Future days are back

RETROMAGAZINE WORLD

NUMERO 48 - LUGLIO 2024 - WWW.RETROMAGAZINE.NET

ISSN 3103-697X

SEGA MASTER SYSTEM

ATLANTEAN SOTTO I MARI CON IL PC ENGINE

PROGRAMMAZIONE - STORIA - INTERVISTE - RECENSIONI - APPROFONDIMENTI

L'importante é partecipare, ma noi vogliamo vincere!

Benvenuti a questo ennesimo numero di RetroMagazine, dove la nostalgia e la tecnologia vintage si incontrano per offrirvi un viaggio affascinante nel passato dell'informatica. Mentre il mondo si è riunito a Parigi per celebrare le Olimpiadi del 2024, noi cogliamo l'occasione per rammentarvi come i computer del passato hanno influenzato il modo in cui viviamo gli eventi di tutti i giorni.

Le Olimpiadi sono sempre state un palcoscenico mondiale per l'innovazione, non solo nello sport ma anche nella tecnologia. Dal cronometraggio elettronico ai sistemi avanzati di gestione delle gare, i computer hanno giocato un ruolo cruciale nel garantire che questi eventi si svolgano in modo fluido e preciso. Il cronometraggio delle gare sportive ha subito notevoli cambiamenti dagli anni '80 a oggi, evolvendosi grazie alle innovazioni tecnologiche che hanno reso il processo più preciso, affidabile e ricco di dati.

Allo stesso modo, lo sviluppo del software ha beneficiato di questa evoluzione tecnologica che ha permesso l'incontro, anche virtuale, di sviluppatori ed appassionati portando una ventata di innovazione nel modo di approcciare la programmazione. Ne sono esempio le decine di giochi che periodicamente vengono sfornati per le nostre retro-macchine e, in questo specifico numero, le conversioni di Prince of Persia per Plus4 e VIC20. Alzi la mano chi avrebbe immaginato possibile 40 anni fa questo piccolo miracolo...

Voglio inoltre invitarvi a riflettere su come il retrocomputing non sia solo una celebrazione della tecnologia del passato, ma un modo per comprendere meglio l'innovazione continua, analizzando come i principi fondamentali della programmazione e del design hardware degli anni '70 e '80 abbiano gettato le basi per le soluzioni avanzate che vediamo oggi.

Mentre Parigi ospita i Giochi del 2024, noi celebriamo la resilienza e la creatività delle menti che hanno plasmato il mondo del retrocomputing.

Speriamo che questo numero continui ad ispirarvi a riscoprire il fascino dei computer del passato e a riconoscere il loro impatto duraturo sull'innovazione tecnologica contemporanea.

Francesco Fiorentini

SOMMARIO

◊ Sega Master System — Pag. 3
◊ Tiny Disk II — Pag. 11
◊ Beige è bello! THE A400 Mini... — Pag. 18
◊ Compilare con Libdragon — Pag. 22
◊ PunyInform - Tutorial 3 — Pag. 24
◊ Da numero a cifre — Pag. 31
◊ Grandi fattoriali — Pag. 34
◊ “Cielo... mio marito!” — Pag. 36
◊ Hare Basic - 10 Volte più veloce — Pag. 42
◊ Quest for the Golden Chalice (Coleco) — Pag. 43
◊ Timo's Castle (C64) — Pag. 44
◊ The Key (CPC) — Pag. 46
◊ Lyle in Cube Sector (MD) — Pag. 48
◊ Atlantean (PCE) — Pag. 50
◊ Dicing Knight Period (Wonderswan) — Pag. 52
◊ Genesis Dawn of a New Day (ZX) — Pag. 54
◊ Goldorak (GX4000) — Pag. 56
◊ Cecconoid (Amiga) — Pag. 58
◊ Mikie (Atari 8bit) — Pag. 60
◊ Roguecraft (Amiga) — Pag. 62
◊ Robo Tito (Atari 2600) — Pag. 65
◊ The Heart of Salamanderland (CPC) — Pag. 66
◊ Formula V20 1985 (VIC20) — Pag. 68
◊ Princess Paloma's Rescue (MSX2) — Pag. 69
◊ Prince of Persia (Plus4) — Pag. 70
◊ Prince of Persia (VIC20) — Pag. 71
◊ Pocket Bomberman (GB/GBC) — Pag. 72
◊ The Amiga Arcade Classic CD32 (Amiga) — Pag. 73
◊ Marathon (Mac/Linux) — Pag. 74
◊ Kien (GBA) — Pag. 76
◊ 3D Pool (C64) — Pag. 77

Hanno collaborato alla stesura di questo numero di RetroMagazine World (in ordine sparso):

- Carlo N. Del Mar Pirazzini
- Daniele Brahimi
- Francesco Fiorentini
- Giulio Fieramosca
- Roberto Del Mar Pirazzini
- Giampaolo Moraschi
- Leonardo Miliani
- Takahiro Yoshioka
- Barbara “Morgana” Murgida
- Eugenio Rapella
- Diamanti Maurizio
- Marta Rossmann
- Gianluca Girelli
- András Vajda
- Fredrik Ramsberg

- Immagine di copertina: **Giuseppe Mangini**
- Layout di copertina: **Carlo N. Del Mar Pirazzini**

Sega Master System

di Leonardo Miliani

Scottata non solo dal fatto che Nintendo abbia presentato il suo Famicom lo stesso giorno del lancio della sua console SG-1000 ma anche per via dello scarso successo commerciale che sta riscuotendo nei confronti della rivale, Sega decide di rivedere l'hardware della sua console e presenta un prodotto ad 8 bit dalle alterne fortune, il Sega Master System (fig. 1). Dominatore dei mercati PAL, esce sconfitto in quelli NTSC ma, nonostante ciò, la console serve come sprone al costruttore giapponese per continuare ad investire nel mercato dei sistemi da gioco domestici per diversi anni a venire.

Dall'SG-1000 al Master System

Come già visto in un precedente articolo, l'SG-1000 nasce su proposta della filiale giapponese di Sega che, nel 1982, vedendo il calo che sta iniziando a colpire il mercato dei giochi arcade soprattutto in Nord America, suggerisce alla casa madre statunitense di puntare sulla produzione di un home computer. Durante lo sviluppo di quello che diverrà l'SC-3000, Sega viene a conoscenza del fatto che Nintendo sta investendo nello sviluppo di una console da giochi domestica, ed il progetto dell'SC-3000 viene rivisto per poter diventare anch'esso una console, l'SG-1000 appunto. L'hardware dei due sistemi è comune, e si basa su una CPU Zilog Z80 con affiancato un chip grafico Texas Instruments TMS9918A ed un chip audio SN76489. Componenti discreti che, proprio per le loro specifiche nella media, non riescono a competere con il Famicom. Eppure, alla fine del 1983, dopo qualche mese dalla sua presentazione, le vendite dell'SG-1000 vanno anche discretamente bene, perché il Famicom presenta dei problemi tecnici e Nintendo non solo richiama le console già vendute ma ritarda anche la distribuzione di quelle a magazzino per correggere questi difetti. Così facendo la console Sega vive qualche mese di gloria dovuto all'assenza di concorrenza, ma poi il prodotto Nintendo non appena ritorna sul mercato inizia nuovamente a macinar vendite. Nel frattempo Sega sta vivendo un momento critico della sua esistenza: West & Gulf, la multinazionale a cui appartiene, decide di disfarsi dei rami societari che non rendono molto e Sega, che in America sta attraversando la famosa crisi dei videogiochi del 1983, non naviga in buone acque, e viene smembrata, con la divisone arcade venduta ad un'azienda rivale mentre vengono momentaneamente tenute la divisione ricerca e sviluppo americana e la filiale giapponese. Le cose purtroppo non

Fig. 1 - Il Sega Master System (foto: Evan-Amos – fonte: Wikimedia Commons)

migliorano e, agli inizi del 1984, viene messa in vendita anche quest'ultima. Fortunatamente, l'ex amministratore David Rosen e l'attuale presidente Hayao Nakayama, con il supporto finanziario del gruppo CSK Corporation, rilevano l'azienda ed iniziano un piano di rilancio. Rivedono l'SG-1000 per renderlo più esteticamente accattivante, risolvendo alcuni problemi progettuali della prima versione, come ad esempio i joystick che non si possono staccare dal corpo principale. L'SG-1000 Mark II viene messo in commercio a metà del 1984 ma, anche in questo caso, le vendite non vanno bene. Nintendo ha attuato un'aggressiva politica commerciale legando a sé gli sviluppatori di videogiochi: chi vuole pubblicare giochi per il Famicom deve rilasciarli in esclusiva per la sua console. Non potendo offrire gli stessi giochi della concorrente, l'offerta di Sega è composta per lo più da conversioni dei suoi arcade. Concettualmente, poi, l'hardware è lo stesso del precedente sistema, basato sul TMS9918A che è oggettivamente inferiore al chip video del Famicom: quest'ultimo offre molti più sprite, oltretutto multicolore, ed una tavolozza molto più ricca. Sega decide pertanto di rivedere il progetto del TMS9918A e ne tira fuori un chip che è capace di 32 colori contemporanei sullo schermo da una tavolozza di 64. Gli sprite gestibili passano a 64, ognuno con 16 colori. Il chip audio è lo stesso dell'SG-1000 ma, per risparmiare spazio e componenti, viene integrato direttamente nel chip grafico.

Il 20 ottorbe 1985, due giorni dopo che Nintendo ha messo in vendita negli USA la versione internazionale del Famicom, ribattezzata Nintendo Entertainment Syste (NES), Sega presenta in Giappone il Mark III, evoluzione dell'SG-1000 Mark II con cui condivide ben poco, essendo una console completamente diversa (fig. 2).

Sebbene dotata di caratteristiche tecniche superiori, la console non riesce ad imporsi sul NES sopratutto per le citate esclusive che Nintendo riesce a strappare ai produttori di videogiochi. Nonostante le non brillanti vendite in patria, Sega decide comunque di esportare la console pensando che riesca a fare meglio sui mercati internazionali di quanto stia facendo in Giappone. Per operare al meglio sul prolifico mercato statunitense, viene fondata una succursale denominata Sega of America al cui comando viene posto Bruce Lowry, che viene direttamente

Fig. 2 - Il Sega Mark III, evoluzione dell'SG-1000 Mark II e progenitore del Master System (foto: Muband – fonte: Wikimedia Commons)

da Nintendo of America, dove ha ricoperto il ruolo di vice-presidente del reparto vendite. La scelta del nome aziendale è anche opera sua: egli infatti reputa che "Sega" e "America" siano due parole che stanno bene insieme, e, poi, venendo da un'azienda che si chiama Nintendo of America, la scelta sembra scontata. Sega of America si occupa inizialmente della revisione estetica della console per adattarla ai gusti occidentali: viene abbandonato il case bianco a favore di uno più scuro, dotato anche di plastiche più lisce e con angoli più pronunciati, ripulendo anche l'aspetto spigoloso della parte superiore e imprimendoci una grafica che ricorda quella dei lettori di musicassette.

Alla console viene assegnato il nome di Master System mediante un sondaggio fatto fra i dipendenti stessi di Sega of America. L'avallo definitivo arriva comunque dal presidente della casa madre giapponese Okawa, che lo approva anche per via del richiamo al fatto che in una qualunque competizione c'è sempre solo un "Master", colui che si impone su tutti gli altri, e ciò viene visto come benaugurante per i destini della console.

Commercializzazione

La console viene presentata al CES di Chicago che si tiene nel mese di luglio del 1986 e messa in vendita nel mese di settembre dello stesso anno. La confezione, bianca e con una trama a quadretti, contiene la console, pomposamente chiamata "Power base", una pistola ottica, due controller ed una cartuccia multi-gioco, il tutto al prezzo di 200 dollari. Sega investe circa 15 milioni di dollari per promuovere la console ma i risultati non sono quelli sperati: alla fine della stagione natalizia, quando la console è in promozione a circa 150 dollari per contrastare le vendite del NES, Sega ha venduto circa 250.000 unità del Master System, meno di un quarto di quanto ha venduto Nintendo. Nonostante Sega spinga sulla qualità dei giochi con lo slogan "la grafica sulla scatola è la grafica sul video", ad indicare che ciò che si vede sulle confezioni dei giochi non sono immagini ritoccate ma il vero aspetto dei titoli così come appaiono sullo schermo, il NES continua a vendere molto di più per via degli accordi di esclusiva di Nintendo ed anche per la poca offerta di giochi da parte di altri sviluppatori. In pratica, in Nord America, i giochi per la console sono prodotti, oltre che da Sega, solo da Activion e Parker Borthers. Per spingere le vendite viene assunto un ex-astronauta della NASA, Scott Carpenter, un po' come Commodore ha fatto qualche anno prima prendendo come volto immagine William Shatner, il capitano Kirk di Star Trek, per pubblicizzare il suo VIC-20. Cercando di incrementare le vendite, Sega of America si accorda con la catena di giocattoli Tonka cedendo l'esclusiva sulla commercializzazione negli USA, sperando che le vendite della console, acquistabile anche nei comuni negozi di giocattoli, possano risalire. Purtroppo Tonka non ha esperienza nella gestione delle vendite di dispositivi videoludici elettronici ed i numeri continuano a languire. Alla fine del 1987 Sega mette in vendita il Master System anche in Giappone, ma anche qui le cose continuano a non andar bene, non riuscendo a contrastare il NES.

Per il Master System va molto meglio in altri mercati. In Europa la console arriva a metà del 1987, inizialmente in Gran Bretagna, Francia e Germania, ma con un avvio della commercializzazione segnato da un incidente di percorso. A causa della lentezza di Sega nel soddisfare i preordini, i distributori locali accusano forti perdite economiche e recidono i contratti. Agli inizi del 1988 Virgin Mastertronic prende in mano la distribuzione su tutto il mercato europeo e spinge la console facendo leva sulle sue superiori capacità grafiche rispetto al Commodore 64 ed al Sinclair Spectrum. Forte anche della non buona penetrazione sul suolo europeo del NES, il Master System si afferma velocemente come il sistema da gioco più venduto: nel 1990 Virgin Mastertronic vende poco meno di 1 milione di Master System. Forte di questi numeri, diversi sviluppatori di giochi europei decidono di pubblicare i loro titoli per la console di Sega, con il risultato che il parco giochi del Master System versione PAL batte quello NTSC: sono circa 100 i giochi prodotti negli USA, contro i 300 prodotti in Europa.

La console viene presentata anche in Sud Corea, dove riceve una buona accoglienza e resta la console più venduta fino almeno al 1993. Anche in Australia i numeri sono molto buoni. Alla fine del 1989 il Master System viene distribuito anche in Brasile, ad opera di TecToy, che lavora molto bene commercializzando la console con successo, spingendola non solo con la pubblicità ma anche realizzando localizzazioni in portoghese dei giochi per la console.

Nel 1988 viene presentato in Giappone il Mega Drive, la prima console Sega a 16 bit, e subito tutti gli sforzi della società si concentrano sulla nuova arrivata. Il Master System cade presto nel dimenticatoio: l'ultimo gioco pubblicato è del 1989, poi un lento declino fino al 1991, quando viene tolta dal mercato. Nel 1992 sparisce anche dal Nord America, senza lasciare troppa traccia di sé. Nel 1994 esce di scena anche dalla Corea mentre in Europa,

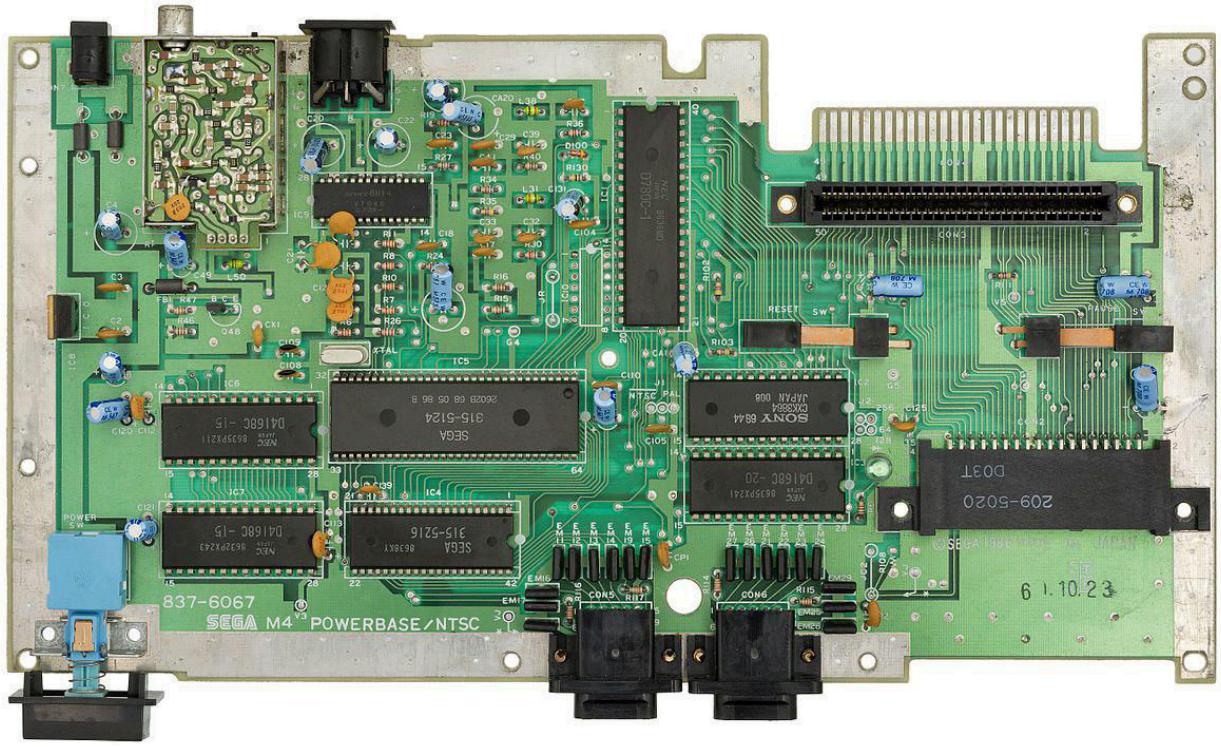


Fig. 3 - Scheda madre del Sega Master System: si notano la CPU Z80 (il chip centrale posto verticalmente), il VDP (il grosso chip leggermente a sinistra rispetto al centro). Sul lato destro la porta Sega Card (in basso) e quella per le cartucce (in alto) (foto: Evan-Amos – fonte: Wikimedia Commons)

grazie ai giochi che continuano ad essere regolarmente pubblicati, resiste fino al 1996. Il Brasile resta un caso a parte: terminata la commercializzazione ufficiale, TecToy ha continuato a produrre, ad intervalli regolari, varie console compatibili con il Master System e, recentemente, anche un dispositivo compatibile sia con il Master System che con il Mega Drive.

Caratteristiche tecniche

Sviluppata per risultare migliore del NES, la console riesce ad assolvere questo compito in maniera egregia, offrendo una grafica con più sprite e colori della controparte Nintendo. Come detto, il Video Processing Unit (VDP), il cuore grafico del Master System (fig. 3), deriva direttamente dal TMS9918A, un chip di Texas Instruments ben noto agli amanti dei computer anni '80 avendo equipaggiato sia il TI-99/4A sia tutti i computer della serie MSX. Partendo da questa base, i progettisti Sega hanno tirato fuori un'unità capace di visualizzare un'immagine con molti più colori: mentre il TMS9918A è limitato a 16 colori (anzi 15, perché il 16° in realtà è un "non colore", una trasparenza) il VDP offre una tavolozza di ben 64 colori, di cui 32 visibili contemporaneamente sullo schermo, secondo una ripartizione particolare: 16 colori sono disponibili per le "tile" (le tessere, o caratteri, di 8x8 pixel che compongono lo sfondo) mentre altri 16 sono a disposizione per gli sprite. La tavolozza è indicizzata ed i 32 colori sono scelti fra i 64 disponibili e numerati da 0 a 31. In realtà c'è una piccola precisazione da fare: mentre per i colori dello sfondo si può scegliere fra i colori per le tile (dallo 0 al 15) oppure fra quelli per gli sprite (dal 16 a 31), per questi ultimi si può scegliere solo fra i colori a loro riservati (dal 16 al 31).

La risoluzione è di 256x192 pixel ma può salire a 256x224 e, nei sistemi PAL anche a 256x240: in quest'ultimo caso non si ha il classico bordo intorno all'immagine. Il VDP gestisce un massimo di 488 tile differenti per lo sfondo, che possono essere riflesse in orizzontale o verticale. Inoltre, le tile possono apparire sia davanti che dietro agli sprite. Questi ultimi possono essere 64 contemporaneamente sullo schermo (il doppio di quelli offerti dal TMS9918A originale), per un massimo, però, di 8 per singola riga video. Rispetto a quelli gestiti dal chip da cui deriva, gli sprite del VDP sono multicolore e le loro dimensioni possono essere di 8x8 oppure 8x16 pixel: esiste anche la possibilità di ingrandirli raddoppiandone le dimensioni: essi passano, rispettivamente, a 16x16 e 16x32 pixel. Come le tile dello sfondo, anche gli sprite attingono i dati a blocchi di 8x8 pxiel dalla memoria. Ciò comporta che più sprite con dettagli diversi si hanno più memoria si

Fig. 4 - Una Sega Card con il gioco Hang-On (fonte: fullyretro.com)

toglie alle tile dello sfondo, e viceversa. Infine, il chip gestisce anche lo scrolling verticale e orizzontale.

Il chip audio è integrato nello stesso silicio del VDP: si tratta del core audio dell'SN76489, capace di 3 voci ad onda quadra e di un generatore di rumore. Niente di esaltante, ma nel complesso un discreto processore audio molto diffuso, essendo usato sul TI-99/4A, sul BBC Micro, sul ColecoVision e su diversi giochi arcade. Nella versione originale giapponese della console, la Mark III, è presente un ulteriore chip audio, lo Yamaha YM2413, per la generazione di audio FM.

La CPU è la stessa Zilog Z80 dell'SG-1000, con clock di 3,58 MHz. La console offre 8 KB di RAM per la memorizzazione dei dati del gioco in esecuzione e 16 KB di VRAM, esclusivi del VDP. I giochi sono distribuiti su 2 supporti differenti: le classiche cartucce ROM e le Sega Card, un formato proprietario introdotto con l'SG-1000 (fig. 4). Queste ultime sono di dimensioni più contenute e più economiche da produrre, risultando in giochi con un costo di acquisto inferiore rispetto a quelli distribuiti sulle classiche cartucce ma, per contro, offrono significativamente meno spazio di archiviazione: mentre la capacità di una Sega Card va da 4 a 32 KB, quella di una cartuccia ROM va da 16 a 512 KB. Una curiosità: come per altri produttori, anche Sega ha distribuito i suoi giochi riportando la capacità del supporto in bit invece che in byte, in modo che sembrino più grandi e diano così l'idea di contenere più dati del gioco: si hanno così supporti con capacità, ad esempio, di 256 Kbit (32 KB), 1 Mbit (128 KB), 2 Mbit (256 KB) oppure 4 Mbit (512 KB).

Esteticamente la console si presenta come un parallelepipedo di colore nero, largo e basso. Anteriormente ci sono 2 prese per i controller ed il pulsante di accension mentre posteriormente ci sono l'uscita video composita, l'uscita RF per il collegamento ad un comune TV, il selettore del canale televisivo su cui far uscire il segnale e l'ingresso dell'alimentazione. Sul lato superiore ci sono il vano per l'alloggiamento delle cartucce ROM nonché la porta per le Sega Card oltre che i pulsanti di reset e di pausa del gioco. I controller di gioco sono rettangolari, con una crociera direzionale con possibilità di montare una piccola astina per trasformarla in un piccolo joystick, e due pulsanti.

Accessori e revisioni

Per il Master System sono stati prodotti diversi accessori. Quelli più famosi sono gli occhiali SegaScope 3-D (fig. 5) e la pistola ottica Light Phaser (fig. 6). I primi sono degli speciali occhiali che permettono, se abbinati a giochi sviluppati appositamente, una visione steroscopica. Gli occhiali si collegano alla console tramite la porta delle Sega Card e per essi sono stati sviluppati solo 8 giochi, tra cui "OutRun 3D" e "Zaxxon 3-D". La Light Phaser è una pistola ottica realizzata riproducendo il profilo dell'arma usata nell'anime giapponese Zillion, ed è stata usata in 13 giochi, tutti pubblicati al di fuori del Giappone. Non mancano le rivisitazioni delle classiche periferiche di gioco, come il Sega Control Stick, un joystick in stile arcade, oppure il Sega Sports Pad (non rilasciato in Europa), un trackball studiato per i giochi sportivi ma usato solo in 3 titoli ("Sports Pad Football", "Sports Pad Soccer" e "NFL Sunday Ticket").

Sega ha rilasciato anche una versione portatile della console, il Sega Game Gear (fig. 7): è stato presentato in Giappone a fine 1990, poi nel 1991 in Europa e Stati Uniti e infine l'anno successivo in Australia e Nuova Zelanda. Si tratta di un dispositivo a batterie con un piccolo schermo LCD a colori di 3,2 pollici di diagonale: rispetto al Master System, la risoluzione è di 160x144 pixel ed i colori sono sempre 32 contemporaneamente su schermo ma il suo

Fig. 5 - I Sega Scope 3-D
(foto: Boffy B – fonte: Wikimedia Commons)

VDP ha una tavolozza espansa a ben 4.096 diverse tonalità. Tramite un adattatore denominato Master Gear è possibile usare i giochi del Master System sul Game Gear, grazie al fatto che condivide praticamente lo stesso hardware della console.

Nel 1990 la console viene rivista e, per contenerne i costi di produzione, viene ridotta di dimensioni e vengono rimosse alcune caratteristiche fra cui l'uscita video composito, il pulsante di reset e la porta per le Sega Card. Viene integrato un gioco, che si avvia quando la console viene accesa senza una cartuccia inserita: per le versioni NTSC c'è "Alex Kidd in Miracle World" mentre per quelle PAL "Sonic the Hedgehog". Viene messa in commercio come Master System II ed incontra un buon favore di pubblico sopratutto in Europa e Brasile (fig. 8).

TecToy, grazie al successo della console sul suolo brasiliano, presenta nel 1992 la Master System III Compact, che in realtà non è altro che il Master System II rimarchiato (fig. 9). La console viene distribuita con una cartuccia multigioco allegata alla confezione. Sempre TecToy ha prodotto, sempre per il solo mercato brasiliano, la Master System Super Compact, di dimensioni più ridotte e dotata di un trasmettitore radio per inviare il segnale al televisore. Nel

Fig. 7 - Il Sega Game Gear
(foto: Evan-Amos - fonte: Wikimedia Commons)

Fig. 6 - La Light Phaser, progettata sulla pistola laser dell'anime "Zillion"
(foto: Evan-Amos – fonte: Wikimedia Commons)

2008 TecToy ha rilasciato infine una versione del Master System III basata su un SoC con 131 giochi integrati.

Giochi

Come già ampiamente ribadito in precedenza, Nintendo ha saputo garantirsi l'esclusiva della pubblicazione dei giochi per la sua console. Questo ha tagliato fuori... dai giochi diverse altre console, fra cui appunto il Master System di Sega, che su diversi mercati non è riuscita a reggere il confronto con la rivale, come in Giappone e in Nord America, da sempre due dei più prolifici per i videogiochi. Nonostante ciò la buona diffusione avuta su altri mercati, come quello europeo o quello brasiliano, ha invogliato diversi sviluppatori a produrre titoli specifici per il Master System. Soprattutto in Europa il fenomeno si è fatto sentire con una produzione così ricca che ha surclassato di gran lunga quella nordamericana: molti giochi sviluppati in Europa sono infatti esclusive pubblicate solo per la console in versione PAL, come "Asterix" (fig. 11). Come poi Nintendo ha fatto con Mario, anche Sega ha cercato fortuna creando una mascotte e affidando ad essa il ruolo di rappresentante della console stessa. Se almeno inizialmente questo ruolo è stato rivestito in modo non ufficiale da "Alex Kidd", comparso in "Alex Kidd in Miracle World" (fig. 10) del 1986, questo ruolo è stato poi affidato ufficialmente a "Sonic the Hedgehog" (fig. 18), il porcospino blu comparso per la prima volta nell'omonimo titolo del 1991. Oltre ai giochi basati su questi personaggi sono molto famose le conversioni degli arcade di Sega o di altri produttori, come "Bubble Bobble" (fig. 15). Ci sono giochi che hanno saputo più di altri tirar fuori il meglio dal Master System e sono divenuti nel tempo vere icone, ad esempio "Ninja Gaiden" (fig. 13), quelli con Topolino come personaggio protagonista come "Castle Illusion starring Mickey Mouse" (fig. 12) o

Fig. 8 - Il Sega Master System II (foto: Evan-Amos - fonte: Wikimedia Commons)

Fig. 9 - Il Sega Master System III, esclusiva brasiliana commercializzata da TecToy(foto: Neowoikkin2113 - fonte: Wikimedia Commons)

"Wonder Boy III" (fig. 17). Grazie alle capacità grafiche della console, la produzione di giochi è rimasta attiva anche dopo la sua dismissione e la diffusione dei sistemi di nuova generazione, con conversioni di giochi nati per questi sistemi, come "Street Fighter II" o "Prince of Persia" (fig. 16).

Eredità

Nonostante nella sua vita commerciale non abbia fatto registrare numeri da record, il Master System resta comunque una pietra miliare nella storia di Sega. E' la console che ha rafforzato l'immagine dell'azienda giapponese come produttore di console, raccogliendo l'eredità dell'SG-1000 e consolidando la sua presenza nel settore, e preparando il territorio per quel successo di vendite che è stato poi il Mega Drive, nonché introducendo la mascotte Sonic che è divenuta l'immagine stessa dell'azienda. Le sue caratteristiche grafiche hanno lasciato il segno in migliaia di fan tant'è che, a distanza di anni dalla sua dismissione, in alcuni Paesi console messe in vendita in operazioni di amarcord non proprio sicure si sono invece rivelate vere e proprie cacce all'oggetto per cultori della console. Alla fine dei salmi, le 13/15 milioni di console vendute in tutto il mondo e le circa 8 milioni di unità vendute nel solo Brasile sono numeri che esprimono bene l'amore per un oggetto che è rimasto nel cuore di molti.

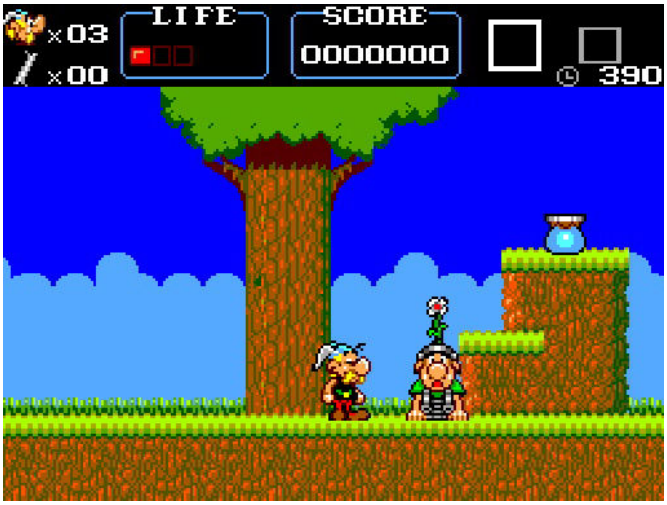


Fig. 10 - Asterix (fonte: Mobygames.com)

Fig. 11 - Alex Kidd in Miracle World (fonte: Mobygames.com)

Fig. 12 - Castle Illusion - starring Mickey Mouse (fonte: Mobygames.com)

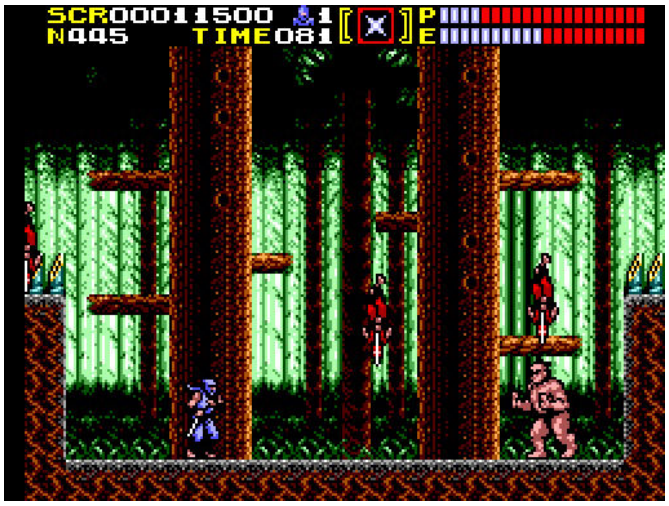


Fig. 13 - Ninja Gaiden
(fonte: Mobygames.com)

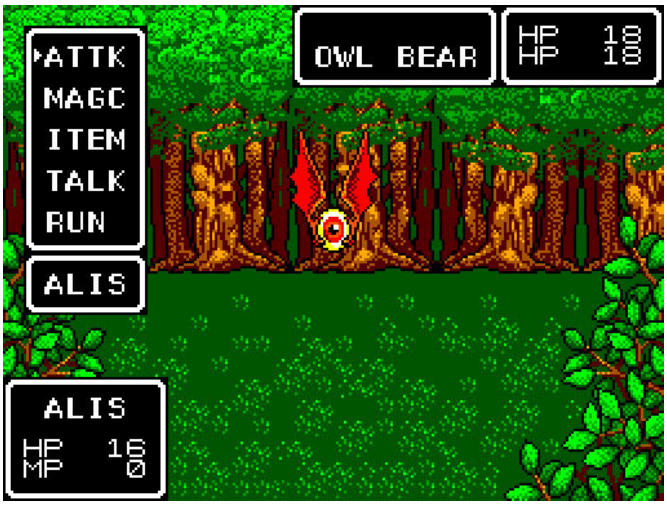


Fig. 14 - Phantasy Star
(fonte: Mobygames.com)

Fig. 15 - Bubble Bobble
(fonte: Mobygames.com)

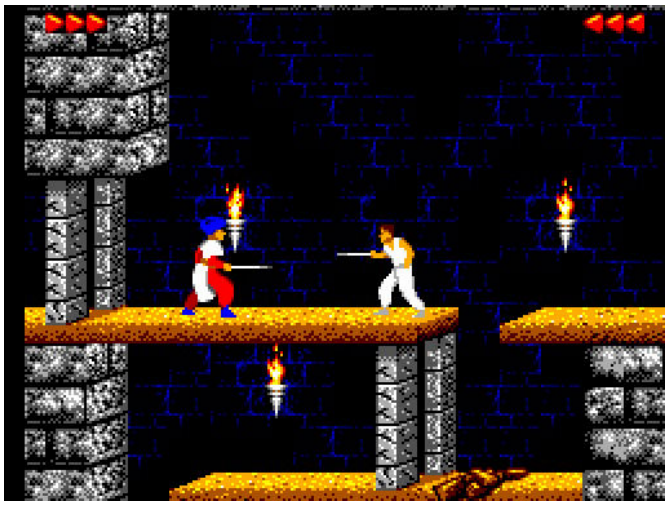


Fig. 16 - Prince of Persia
(fonte: Mobygames.com)

Fig. 17 - Wonder Boy III - The dragon's Trap
(fonte: Mobygames.com)

Fig. 18 - Sonic the Hedgehog
(fonte: Mobygames.com)

Tiny Disk II

di Giulio Fieramosca

Una decina di anni fa sono entrato in possesso di un Apple] [e, salvato prima che venisse gettato. I molteplici problemi - i chip di memoria guasti e il floppy drive meccanicamente guasto - non lo hanno reso molto appetibile per le sperimentazioni, ed è rimasto per anni nel garage della nostra "RetrOfficina", tirato fuori soltanto per le rare esposizioni. Questo fino al 2022 quando mi sono deciso di dedicare un bel po' di tempo al restauro di questa interessante e storica macchina.

Tagliando corto, il floppy drive originale è rotto e non sono riuscito a ripararlo. Il che è un gran dispiacere, visto che buona parte del software per questa macchina circolava su dischetto. Non solo, pur essendo dotata di interfaccia a nastri, in molti casi era necessario precaricare versioni più avanzate di BASIC o di DOS... da dischetto. Mi sono guardato in giro per valutare l'acquisto di uno di quegli onnipotenti emulatori di floppy, come il floppyEmu o il wDrive. Sono belli ma costosi. Il che mi ha spinto a realizzare la mia personale versione di emulatore floppy, con l'intenzione di rendere tutto il progetto disponibile online. Prima di reinventare l'acqua calda ho dato un'occhiata sul web alla ricerca di un qualche progetto libero, ed in effetti mi sono imbattuto in un certo "SDISK II", di cui sono disponibili online sia gli schemi che il sorgente. Ho provato a realizzarlo senza successo, ma non ci ho messo molto impegno: l'hardware non mi convinceva del tutto. In ogni caso il firmware è stato un ottimo ed utilissimo punto di partenza su cui lavorare per il mio progetto.

In tutto questo non avevo la minima idea di come funzionasse un floppy, men che mai quello dell'Apple II. Quindi, per prima cosa ho iniziato a documentarmi per studiare l'elettronica ed il protocollo.

Il protocollo - Organizzazione fisica dei dati

L'involucro morbido dei floppy da 5¼" protegge un dischetto di materiale plastico, ricoperto di un sottile strato magnetizzabile. Le informazioni sono registrate digitalmente applicando un campo magnetico localizzato ad un'area ristretta, che vi resta impresso. La lettura avviene con il processo inverso, captando i cambiamenti nella magnetizzazione del dischetto.

Lettura e scrittura avvengono mediante una coppia di testine, poste a distanza ravvicinata dal dischetto, mentre quest'ultimo è in rotazione. Immaginando la superficie suddivisa in intervalli regolari, si utilizzata una particolare codifica per mappare i bit su disco: il bit "1" si rappresenta con una inversione di polarità e lo "0" con l'assenza di transizioni di polarità.

Il campo magnetico impresso nel dischetto sarebbe troppo debole per poter essere direttamente digitalizzato: per questa ragione è necessario un amplificatore che mantenga costante l'ampiezza del segnale analogico, così da poter leggere sia un disco appena scritto che uno più rovinato. Sfortunatamente, una lunga sequenza di zeri dà un output pressoché costante ed il Drive II tenderà ad "alzare il volume"... arrivando a captare le minuscole variazioni dovute al rumore di fondo. È per questa ragione che la scrittura di una sequenza di bit non può mai contenere più di due zeri consecutivi. Questa è una grossa limitazione

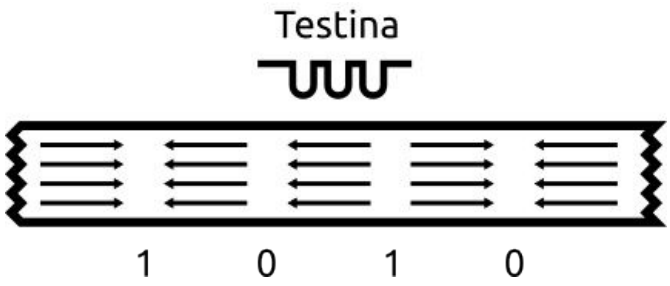


Fig. 1 - Rappresentazione della superficie magnetizzata del dischetto. Ogni inversione di polarità è associata ad un bit "1".

che ci impedisce, per esempio, di trascrivere un testo direttamente sul disco: la lettera A maiuscola, codice ASCII 01000001, sarebbe proibita!

Ma i problemi non sono finiti qui. Durante la lettura, la testina capta un flusso continuo di bit e il Drive II non prevede un mezzo meccanico per segnalare l'inizio di un segmento valido.

Per ovviare a questo inconveniente, il Drive II fu progettato imponendo che un byte valido debba sempre avere un bit "1" in testa. La lettera A maiuscola è a maggior ragione proibita.

Inoltre, per garantire un corretto allineamento, prima di ogni segmento dati valido è anteposta una particolare sequenza di 12 bit, pari a "1111111100", che porta il sistema di lettura ad allinearsi sempre sul primo "1". Questo codice viene anche chiamato "self-synced FF".

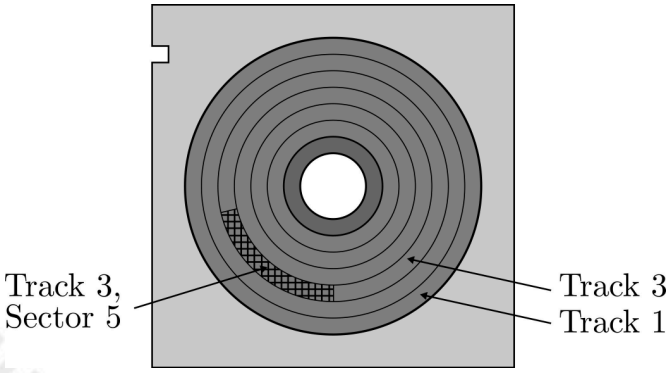
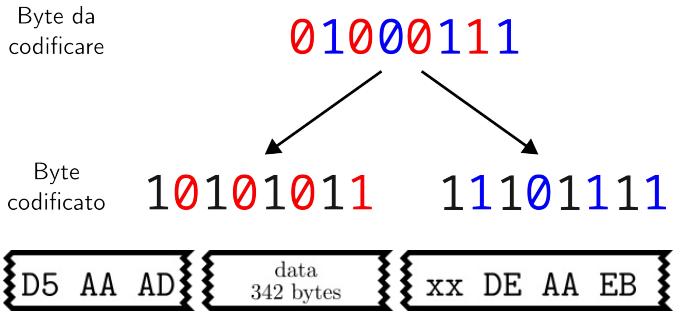


Fig. 2 - Geometria del fisco floppy

Come sui più recenti floppy da 3.5" e sui dischi rigidi, il disco è suddiviso in tracce concentriche e per ogni traccia i dati sono segmentati in settori, così da poter gestire letture e scritture in gruppi più piccoli. Il contenuto informativo di una traccia sarà qualcosa di simile alla figura di seguito.

La particolarità dell'Apple Drive II, che lo differenzia da altri modelli dell'epoca, sta nella estrema semplicità dell'hardware. Per contro, la gestione della geometria del disco è interamente a carico del software.

Dunque, è necessario in qualche modo distinguere un settore da un altro. Per questa ragione ciascun a settore è associato un campo di descrizione, detto "address field". L'address field contiene le informazioni di identificazione del settore all'interno del disco, ad esempio "traccia numero 2, settore numero 7". Questi valori richiedono un byte ciascuno, ma non possono essere trascritti direttamente perché potrebbero non essere rappresentabili su disco. Per questo, ciascun numeretto viene prima codificato con un approccio "odd-even": data la sequenza "10101010", si sostituiscono agli "0" i bit dispari del byte da codificare. A partire dalla stessa sequenza si fa lo stesso con i bit pari, ottenendo due byte distinti. È evidente che, così facendo, il primo bit sarà sempre 1 e non si potranno mai avere 0 consecutivi.



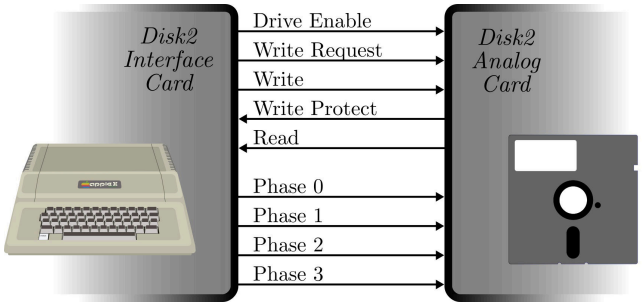
All'address field segue il data field, che contiene l'informazione vera e propria. I dati non sono codificati con la tecnica "odd-even" perché comporterebbe un raddoppio nello spazio occupato. Sono state preferite altre codifiche ad efficienza maggiore, dette "5-and-3" e "6-and-2", ma la loro implementazione trascende dallo scopo di questo racconto, per cui rimando direttamente al manuale "Beneath Apple DOS" per approfondire. Ci è sufficiente sapere che in un settore possiamo immagazzinare 256 byte, che adottando la codifica più efficiente lievitano a 342 byte.

È importante sottolineare che entrambi i fields hanno un prologo univoco di tre caratteri, importanti per distinguerli e decodificarli correttamente. I primi due byte di entrambi i campi, "D5 AA", sono codici riservati che non possono essere utilizzati altrove e fungono da ulteriore sistema di allineamento.

Come accennato, ciascun field è preceduto dalle sequenze di "self-sync FF". Oltre a servire da pattern di sincronizzazione, questi "gap" fungono da buffer per la

scrittura.

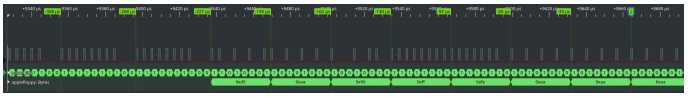
Siccome l'operazione di scrittura è implementata in software ed avviene "alla cieca", queste aree di buffer servono ad avere margine sufficiente per evitare che la riscrittura di un field sovrascriva porzioni di settori adiacenti.



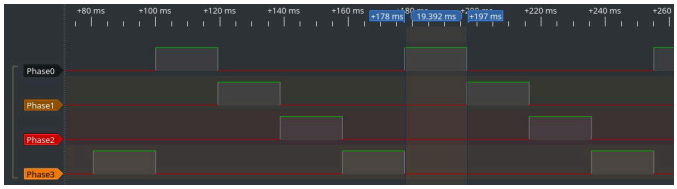
Il sistema del floppy drive dell'Apple II è fondamentalmente composto di due schede: il drive ne contiene una, denominata "analog card", che si occupa di effettuare la conversione dal mondo analogico al mondo digitale; la seconda scheda è contenuta all'interno dell'unità centrale, detta "interface card", e si occupa del coordinamento ad alto livello. Include, fra le altre cose, una ROM per il codice di bootstrap ed una serie di chip necessari a "serializzare" e "deserializzare" i byte da scambiare con il processore. Di fatto, l'emulatore di floppy dovrà spacciarsi per la analog card.

Il drive resta in attesa finché il segnale "drive enable" è disattivato. Attivare il drive significa portarlo in modalità lettura, e da quel momento lungo il filo di read inizieranno a scendere i dati letti dalla testina. La logica di digitalizzazione ripulisce tutto quel che la testina è in grado di captare e restituisce un segnale squadrato costituito da un treno di impulsi, uno per ciascun bit "1" rilevato, e con spaziatura di 4µs.

Il drive commuta in scrittura non appena si attiva anche il segnale di "write request". In questa fase viene accesa la testina di scrittura che col proprio campo magnetico va a sovrascrivere qualsiasi cosa ci fosse in precedenza. Commutare o lasciare invariato il "write" determina il nuovo contenuto del disco, ovvero se scrivere un bit "1" oppure uno "0".



Resta la gestione del posizionamento della testina sulla traccia, di fatto tramite un motore passo passo controllato direttamente tramite le quattro fasi con attivazione sequenziale. Per cambiare traccia in realtà è necessario attivare due fasi consecutive, probabilmente per garantire una maggiore precisione nell'allineamento motore-traccia. Le "tracce intermedie" venivano utilizzate per particolari meccanismi di protezione anticopia, che però non sono stato interessato ad emulare.

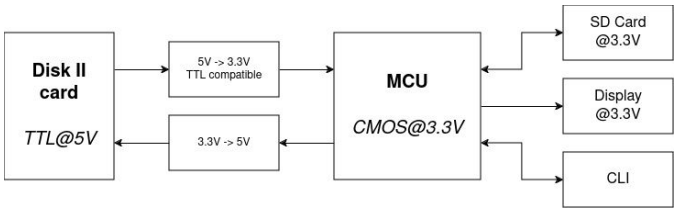


L'emulatore

Per implementare l'emulatore ho seguito le orme del progetto SDISK II ed ho utilizzato soltanto un microcontrollore. Le immagini disco da servire vengono prelevate da una comune SD card formattata in FAT e servite sull'interfaccia disco.

Per scegliere l'immagine e monitorare lo stato del drive ho predisposto in prima battuta una interfaccia a riga di comando via seriale, poi una interfaccia semi grafica tramite display OLED e selettore, che non richiede la presenza di un secondo computer.

La scelta del microcontrollore è ricaduta sull'architettura AVR - per intendersi, la stessa della prima serie Arduino – perché avevo una scheda di sviluppo con una variante più performante da utilizzare per le prime prove. Inoltre, anche l'SDISK II è basato su AVR ed il suo codice è stato un buon punto di partenza.



Tutta la sezione "moderna" opera a 3.3V, un vincolo imposto dalla scheda SD e dal display. L'interfaccia con l'Apple II richiede particolare cautela, non solo per le differenti tensioni operative (5V vs 3.3V), ma anche per gli standard logici non compatibili (LVCMOS vs TTL): anche riscalando le tensioni, le due parti potrebbero non capirsi alla perfezione. Nella prima versione sperimentale mi sono accontentato di usare quel che avevo: traslatori di livello economici ma lenti per scendere verso l'emulatore e qualche buffer per rigenerare il segnale dell'emulatore. Fortunatamente esistono dei circuiti integrati adatti allo scopo che ho inserito nella versione realizzata su PCB.

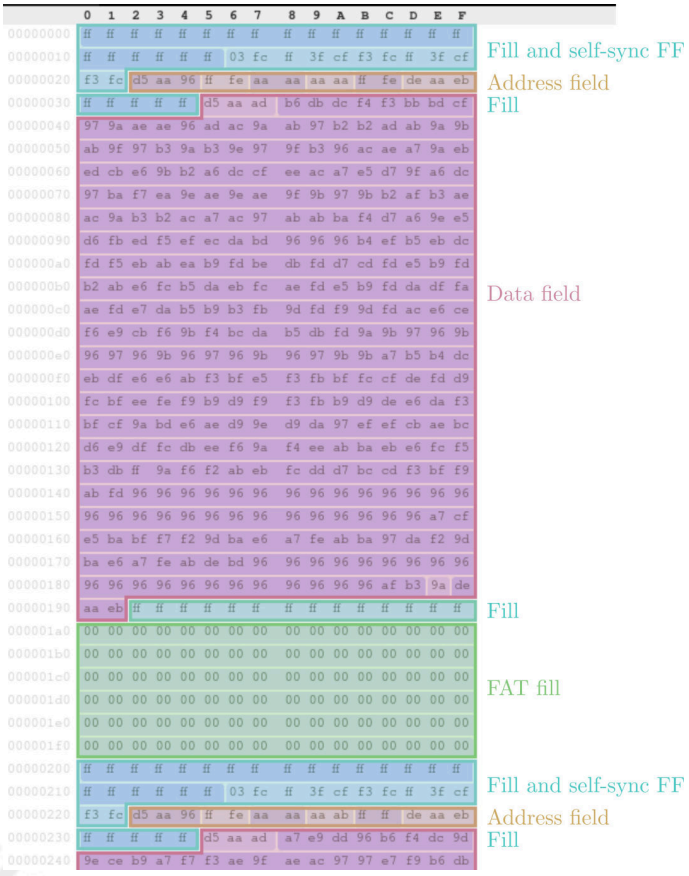
Ho approfittato anche dei LED presenti sulla scheda di sviluppo per avere un feedback visivo dello stato del lettore, per esempio riportando la posizione della testina.

Lettura

Esistono più formati con cui possono essere digitalizzate le immagini disco dell'Apple II, ma per mantenere decenti

le prestazioni dell'emulatore ho mantenuto il formato .NIC utilizzato sull'SDISK II. Si tratta di un file binario che memorizza i bit grezzi fisicamente letti dalla testina del lettore. Il file è strutturato per settori e tracce crescenti. Ho riportato un paio di estratti nella coppia di figure che seguono, evidenziando la struttura dati ed il sopra citato address field.

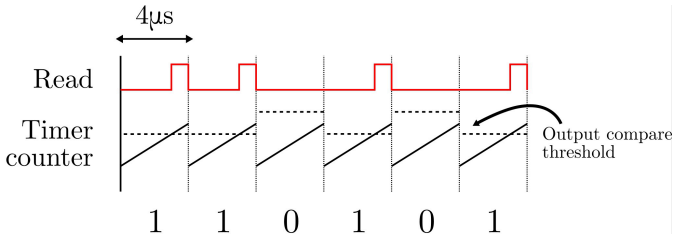
Comprendendo i gap, la dimensione di ciascun settore nel .NIC è di 416 byte. A questi, sono aggiunti 96 byte di fill a 0, così da mantenere ciascun settore allineato a blocchi di 512 byte. Questa scelta è dettata dalla specifica di funzionamento di lettura e scrittura da scheda SD, che richiede di operare a multipli di un blocco, proprio pari a 512 byte.



Data la memoria limitata presente sul micro, non è possibile scaricare tutta l'immagine .NIC in RAM all'inizio della lettura. Piuttosto, noti la traccia ed il settore che devono essere forniti al computer, è sufficiente posizionare l'indirizzo di lettura della scheda SD nel punto desiderato ed, ogni 4 microsecondi, "tirare fuori" un bit, generando o meno un impulso.

Questo può essere svolto interamente a controllo di programma, cosa effettivamente fatta nel progetto originale dell'SDISK II mediante l'uso di alcune routine in assembly. Nella mia implementazione ho voluto delegare la generazione del segnale di lettura ad una periferica hardware presente nel microcontrollore, detta timer PWM. Tale periferica permette di generare autonomamente un segnale ad onda quadra, programmandone il periodo e la frazione di periodo in cui il segnale deve "rimanere basso".

Questa periferica non è altro che un contatore con un singolo pin di uscita associato. Se ne possono programmare il valore massimo - il periodo - ed un valore di soglia. Finché il valore del contatore è minore del valore di soglia, il pin rimane a 0. Superato il valore intermedio, l'uscita viene portata ad 1.



Volendo generare una sequenza di "1" è sufficiente programmare la periferica con un periodo di 4 µs e un tempo di attesa di 3 µs. Impostando il valore di soglia ad un valore superiore al periodo, non avverrà mai il "match" e l'uscita resterà nulla, generando uno "0".

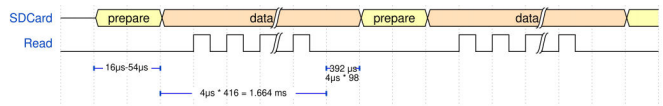
Al termine di ogni periodo il software va ad aggiornare il valore di soglia in accordo con il bit effettivo da trasmettere. Non è più necessario effettuare una attesa attiva perché la periferica PWM è programmata per richiedere una interruzione al termine di ogni periodo. Ad ogni interruzione il programma ha poco meno di 3 millisecondi per leggere un bit dalla scheda SD e decidere se abilitare l'uscita del PWM per l'impulso.

Il Drive II non ha coscienza della suddivisione in settori del disco, per cui la traccia corrente viene comunque letta per intero e trasferita alla scheda di interfaccia. Le operazioni su un settore specifico sono effettuate in software, leggendo ed interpretando ciascun address field per decidere se scartare i dati provenienti da altri settori.

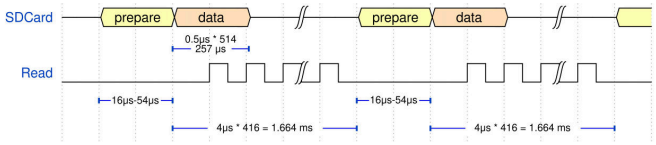
Per l'emulatore, questo si traduce in una specifica di funzionamento molto semplice: finché il segnale di drive enable è attivo, il firmware dell'emulatore cicla la lettura di tutti i settori della stessa traccia.

Il controllo della traccia avviene attraverso i quattro fili di fase PHI0-PHI3, che nel drive originale controllavano direttamente il motore passo-passo di posizionamento

della testina. Nell'emulatore, il cambio della traccia è gestito dal codice principale, non c'è necessità di scomodare interruzioni poiché si tratta di segnali molto lenti, tanto che si riescono ad osservare ad occhio nudo tramite dei LED.
La routine di "controllo traccia" monitora i quattro segnali di fase e incrementa opportunamente il contatore interno di traccia in base alla direzione impartita. Il cambio traccia potrebbe avvenire in qualunque momento, anche nel bel mezzo della lettura di un settore. Per non complicare troppo la logica, ho deciso di non interrompere mai la lettura di un settore e di applicare il cambio traccia solo quando viene fatto il caricamento di un nuovo blocco da SD. Tanto, nel mondo reale, i dati letti durante il cambio di traccia sarebbero comunque invalidi e da scartare.
Il controllo di traccia è incrementale e senza alcun feedback: difatti, il computer effettua, ad ogni accensione, un azzeramento forzato della posizione della testina muovendo il motore alla cieca. È questo il motivo del rumore che fa il disk drive all'accensione. È stato interessante un primo esperimento di lettura in cui non avevo implementato la gestione delle tracce: la prima traccia veniva letta con successo ma, passando alla successiva, il software si accorgeva dell'incongruenza tramite i campi dell'address field. Anche in questo caso veniva tentato l'azzeramento della testina nella speranza di riallinearsi alla traccia giusta, purtroppo senza successo.



Il vincolo temporale più stringente in lettura è imposto dai 4 µs di periodo di bit all'interno dello stesso settore. Invece, non c'è nessuna fretta di servire un nuovo settore. Sia chiaro, le routine software dell'Apple II implementano un timeout che termina la lettura se il settore richiesto non è leggibile entro un certo tempo. Ma questo intervallo di tempo è sufficiente per svolgere un po' di burocrazia. Per servire un nuovo settore è necessario indirizzare correttamente il blocco corretto dalla scheda SD, operazione effettuata a controllo di programma e che richiede qualche decina di µs.
Come dicevo, nel primo approccio non ho fatto uso di buffering di settore. I dati sono letti bit per bit dalla scheda SD ogni 4 µs e serviti immediatamente sulla linea read. La scheda SD impone che i 512 byte di blocco debbano essere letti per intero, per cui al termine del settore restano comunque 96 byte da ignorare, che aggiungono un ulteriore overhead prima di poter iniziare a leggere il settore successivo. Per abbattere questi tempi morti ho sfruttato la presenza della periferica di DMA.
Il DMA, o Direct Memory Access, è un componente del microcontrollore che consente di effettuare scambio dati fra la memoria ed una periferica senza l'intervento diretto del processore. In questo caso la periferica è la SPI che gestisce la comunicazione con la scheda SD.
Con il DMA, l'overhead si riduce al tempo tecnico necessario per impostare sul DMA l'indirizzo del blocco SD da leggere, dopodiché questo verrà caricato in RAM in circa 300 µs. Non importa attendere che il buffer DMA sia stato caricato per intero prima di iniziare a servire i dati sulla linea read, tanto la scheda SD è molto più veloce.
Non mi è sembrato utile ottimizzare ulteriormente la fase di lettura, la scelta di usare il DMA si è rivelata utile più che altro per scrittura e formattazione.



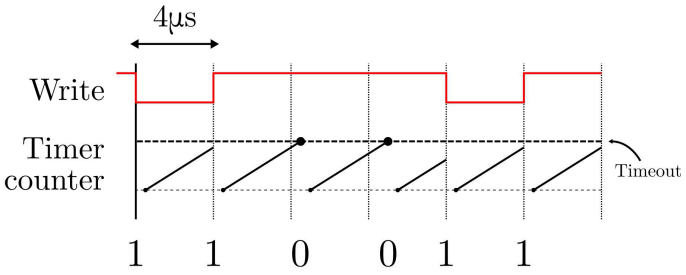
Scrittura

Bisogna differenziare la scrittura di un floppy in due casi distinti, ovvero la sovrascrittura di un settore e la formattazione. Sebbene si tratti in entrambi i casi di operazioni di sovrascrittura, nel primo si va ad alterare il contenuto di un singolo data field, mentre nel secondo caso viene riscritta l'intera traccia, compreso l'address field. La formattazione opera quindi su settori multipli ed ha dei vincoli temporali molto più stringenti, tanto che il primo prototipo di tinyDisk non era in grado di supportare.

Considerando il caso più semplice, la sovrascrittura di un settore, si possono fare le seguenti assunzioni:
-prima di iniziare la scrittura di un settore, la routine Apple attende sempre di ricevere l'address field corretto per potersi allineare sul settore corretto, quindi inizia a sovrascrivere il data field che segue;
- l'address field non viene mai sovrascritto, serve per allinearsi al settore corretto;
- 0xD5 è un byte univoco e riservato, che può essere usato come marcatore di allineamento nella decodifica del segnale write;
- durante la scrittura di un settore non verrà mai cambiata traccia, sarebbe alquanto controproducente.

La strategia per convertire il segnale di write in uno stream binario è semplice: per completare un byte è necessario aver decodificato almeno 8 simboli. Ogni transizione

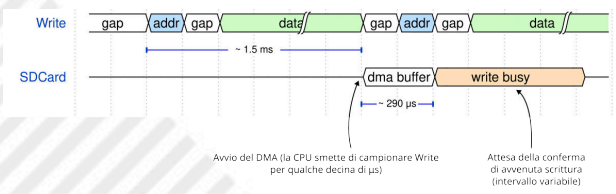
rappresenta un “1”. Lo “0”, ovvero la mancanza di una transizione in uno slot di 4µs, richiede che sia predisposto un timer con periodo leggermente maggiore di 4us, per tenere conto delle minime fluttuazioni del clock dell'Apple II.



La routine software dell’Apple II inizia sempre riscrivendo la sequenza di self-synced FF, che può essere scartata. La prima informazione utile, e fondamentale per potersi allineare, è il prologo del data field, ovvero il byte 0xD5. Rilevato quello, tutti i byte che seguono possono essere salvati in un buffer che sarà opportunamente caricato nel blocco corretto della SD, sovrascrivendo il file .NIC originario. Al termine della scrittura, l’Apple II torna in lettura per allinearsi, eventualmente, con un altro settore da sovrascrivere.

In caso di più settori da sovrascrivere, il mio firmware non è in grado di gestire una nuova scrittura finché non è terminato il “write back” sulla scheda SD. Fortunatamente, questo tempo non è così lungo e l’Apple II ha sufficiente pazienza e non va in errore. Ovviamente è previsto un timeout nel caso in cui non dovesse essere mai rilevato l’address field da sovrascrivere.

Storia ben diversa riguarda la formattazione. Poiché la formattazione va a ridefinire la geometria del disco, in questo caso la traccia viene interamente sovrascritta senza alcuna attesa fra un data field e l’address field del settore successivo. Purtroppo il microcontrollore non ha abbastanza RAM per per memorizzare una traccia - già un singolo settore porta via 512 byte dei 2K disponibili.



La soluzione che mi è sembrata più ragionevole è stata tentare di implementare una pipeline con l’unico buffer di settore disponibile. Il principio è il seguente: registrata la richiesta di scrittura, il buffer viene popolato decodificando il segnale Write, come per la sovrascrittura del settore. Al termine, c'è un breve periodo di grazia rappresentato dalla scrittura del gap fra settori, che il firmware sfrutta per istruire la periferica DMA a trasferire il buffer in scheda SD. A questo punto la scrittura in SD Card può procedere autonomamente senza l'intervento della CPU, che è libera di decodificare il nuovo settore in arrivo.

Pur avendo un singolo buffer è raro che la lettura del nuovo settore sovrascriva una parte non ancora trasferita sulla memoria di massa, perché sussiste un fattore x8 fra il data rate dell'SPI in uscita e del Drive II in ingresso.

La criticità temporale, in effetti, risiede nel tempo di finalizzazione della scrittura in scheda SD: queste memorie, dopo aver immesso il buffer tramite SPI, richiedono un tempo ulteriore per effettuare la cancellazione fisica e la sovrascrittura, tempo che varia in base al modello e all'età del supporto di memoria.

Questo ritardo aggiuntivo ha dato qualche grattacapo, ma si può mitigare forzando il salvataggio dei settori di una stessa traccia in aree fisicamente contigue della scheda SD - di fatto imponendo una blanda non frammentazione dei file .NIC - e permettendo una ottimizzazione nelle operazioni fisiche di cancellazione e sovrascrittura.

Giudizio finale e riferimenti

L'obiettivo di sostituire il drive originale con un emulatore è stato raggiunto con successo. L'ho validato sul campo provando una serie di immagini disco di software "da esposizione": giochi, applicativi, demo... in lettura non ho avuto problemi degni di nota.

Lo stesso vale per la scrittura, testata sia salvando codice BASIC dal DOS, sia modificando arbitrariamente il contenuto dei settori tramite il software CopyII+. La formattazione meriterebbe ancora delle indagini perché, sporadicamente, fallisce quando si ha a che fare con schede SD particolarmente lente o vissute.

Nonostante le semplificazioni, che limitano la fedeltà di emulazione, il progetto si è rivelato una sfida ardua ed averlo realizzato in casa è stato utile per imparare qualcosa di nuovo. Ho provato a riassumere i tratti più salienti di questo progetto senza scendere eccessivamente nel dettaglio sperando di suscitare curiosità ed interesse sull'argomento. Lascio ai più volenterosi la possibilità di approfondire direttamente sul codice del progetto rilasciato con licenza libera, che sia con l'obiettivo di realizzarne una propria versione o proporre modifiche e miglioramenti.

Riferimenti

Repository del progetto:
https://github.com/GLGPrograms/tinyDiskI

SDISK II, home page del progetto:
https://tulip-house.ddo.jp/digital/SDISK2V1/english.html

Beneath Apple DOS:
https://mirrors.apple2.org.za/Apple%20II%20Documentation%20Project/Books/Beneath%20Apple%20DOS.pdf

The Amazing Disk II Controller Card, descrizione del controller disco a cura del progettista del "Floppy Emu":
https://www.bigmessowires.com/2021/11/12/the-amazing-disk-ii-controller-card

Understanding the Apple II:
https://archive.org/details/Understanding_the_Apple_II_1983_Quality_Software/page/n19/mode/2up

Fig. 3 - Apple IIe computer (enhanced version)
(foto: Bilby – fonte: Wikimedia Commons)

Beige è bello!
THE A400 Mini, un gradito ritorno

di Carlo Nithaiah Del Mar Pirazzini

Dopo un'ondata di micro-console che include NES, SNES, Megadrive, PC Engine e il ritorno di micro-home computer come C64 e Amiga, ora mettiamo mano su quello che deve essere l'esperimento più interessante fino ad ora: l'Atari 400 Mini.
Basato sulla gamma di computer domestici 8bit di Atari, che arrivò sul mercato nel 1979, questa piccola versione è un "perla" realizzata con una collaborazione tra Atari, Plaion e Retro Games ltd.
Al momento Atari è impegnata in una sorta di viaggio nel passato, avendo rilasciato una raccolta dei suoi giochi per console moderne e l'Atari 2600+, una replica (quasi) a grandezza naturale della sua console di successo di inizio anni 80.
Tuttavia, l'Atari 400 è un'entità meno riconoscibile e potrebbe potenzialmente essere più difficile da vendere. Vale la pena dargli un'occhiata? Scopriamolo.

Una breve storia

Uscito nel 1979 insieme all'Atari 800, l'Atari 400 offriva un'esperienza audiovisiva più avanzata rispetto ai suoi rivali grazie all'uso innovativo dell'architettura del coprocessore, una novità assoluta per un computer domestico.
Mentre entrambe le macchine utilizzavano la stessa CPU MOS 6502, il 400 era posizionato sul mercato come il più conveniente dei due e aveva una tastiera a membrana e un singolo slot per cartucce (l'800 aveva una tastiera meccanica e due slot per cartucce, oltre a Ram espandibile e tanto altro).
Nonostante fossero stati lanciati come personal computer, l'Atari 400 e 800 erano supportati da un'ampia gamma di giochi, come Star Raiders del 1980 vera e propria Killer Application per la macchina.
Anche la neonata Electronic Arts supportava attivamente il sistema; l'iconico MILE di Danielle Bunten Berry fece il debutto sulla gamma ad 8bit di Atari e sfruttò pienamente le porte controller multiplayer.

Con l'avanzare del tempo, Atari rinnovò la sua gamma con il 1200XL, 600XL e 800XL e, dopo che Sam Tramiel rilevò l'azienda e la rinominò Atari Corporation, la serie XE continuò la discendenza.
Rispetto all'Atari VCS/2600 da 30 milioni di vendite, il duo 400/800 ne vendette circa quattro milioni di unità, non un successo commerciali in termini comparativi, ma comunque un successo relativamente elevato per l'epoca.

ATARI 400 mini – Design e Controller

Il 400 mini sembra una copia sputata del vero computer,

Fig. 1 - La piccola "console" e il classico Joystick CX40

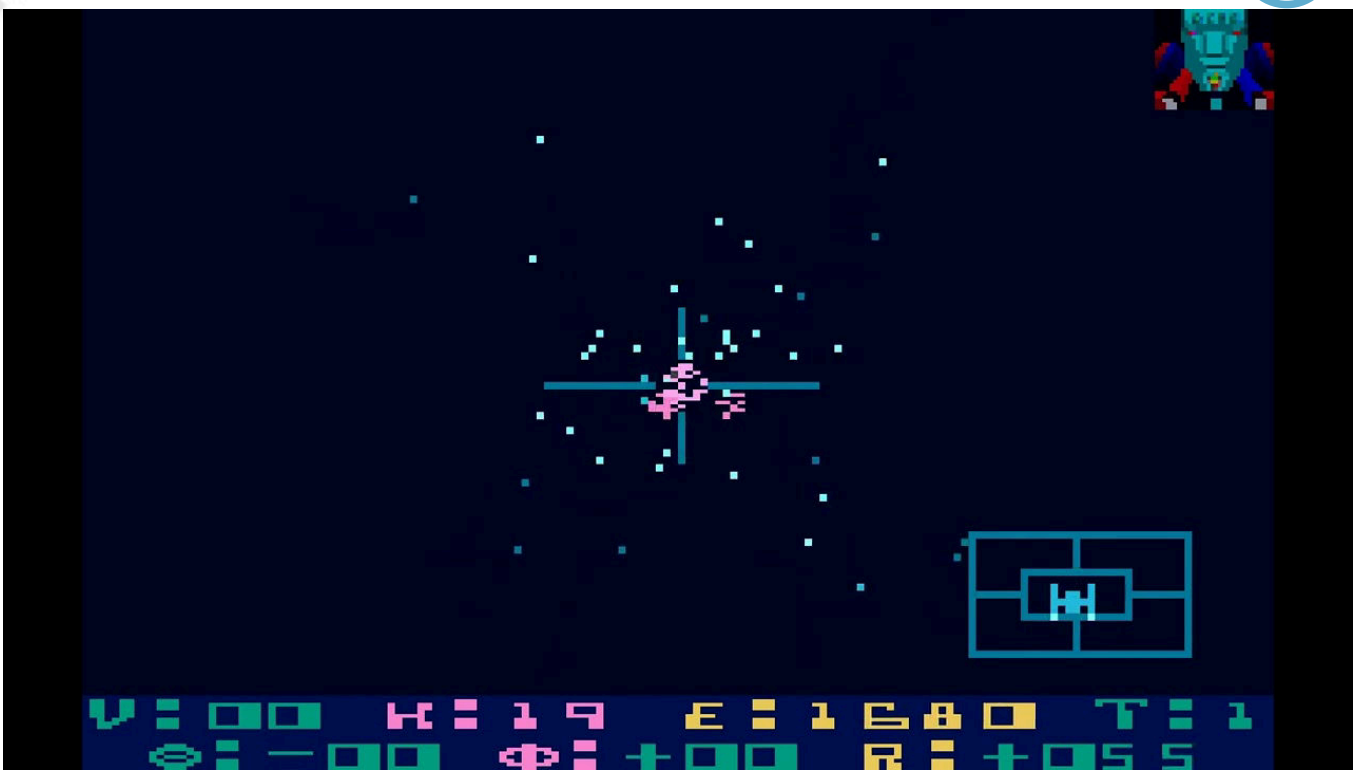


Fig. 2 - Star Raider: la Killer application sui sistemi ATARI 8Bit

con il suo design angolare ispirato a Star Wars, la tastiera a membrana, quattro tasti quadrati gialli e il LED di alimentazione rosso.
L'unica differenza visiva è la scritta "THE400" che sostituisce "Atari 400" nella parte anteriore dello slot pseudo-cartuccia. Il sistema misura 2,0x6,0x5,3 pollici e pasa davvero poco più di 200 grammi. La tastiera è ovviamente non funzionante, come tipico per le emulazioni di console in miniatura.
Il pannello frontale sotto la tastiera contiene quattro porte USB con funzioni da porte Joystick, replicando il computer originale (il che significa che è possibile giocare a MULE in quattro).
Sul retro abbiamo una quinta porta USB, una porta HDMI e una porta USB-C per l'alimentazione. Ovviamente troviamo anche il pulsante di accensione.

Tutti i sistemi ad 8bit Atari hanno tre chip coprocessore personalizzati oltre alla CPU 6502: ANTIC, GTIA e POKEY.

Fig. 3 - Et voilà... Il The 400

Fig. 4 - Il CXStick in dotazione in tutta la sua "turgidezza"

Per mostrare la potenza di questi chip in emulazione, Il 400mini ha un processore AllWinner H3 (in pratica un Cortex A7 quad-core), 256 gb di Ram e 128 gb di spazio di archiviazione, più che sufficienti per salvare i giochi e impostare al meglio il sistema.

Il joystick si chiama TheCXStick e assomiglia al CX40 originale tanto amato da tutti noi. Devo ammettere che è un ottimo stick, con movimenti precisi e nessun clic di troppo. Anche il pulsante di fuoco ha un buon fealing. A differenza dell'originale, questo ha ovviamente un connettore USB all'estremità del cavo. Ha anche un set di pulsanti supplementare (sono 7), ben inseriti nel design e poco invasivi. Quattro sono tasti direzionali nascosti sotto le marcature arancioni. La base ha un pulsante sulla spalla nascosto nell'angolo in alto a sinistra. I pulsanti Black Menu e Home sporgono leggermente dal pannello posteriore.

Nella confezione si trovano anche un cavo HDMI beige, un cavo alimentazione USB-C e il manuale di avvio rapido. Manca l'adattore CA per collegare il cavo USB-C.

Parliamo di giochi

Il sistema è dotato di 25 titoli. Molti sono classici senza tempo, tra cui Bristles, Bruce Lee (rinominato Lee), Capture the Flag, Fip and Flop, Miner 2049er, Star Raiders, The Seven Cities of Golds.

L'Atari 8bit era una macchina perfetta per conversioni di classici arcade e qui abbiamo l'imbarazzo della scelta: Asteroids, Battlezone, Berzerk Centipede, Crystal Castles, Millipede e Missile Command sono perfetti su queste macchine!

Mancano alcuni classici per mancanza di diritti (vedi i titoli Nintendo come Mario Bros, Donkey Kong).

Sono 25 titoli scelti con cura e molto giocabili e sulla pagina https://retrogames.biz/games/the400-mini/ troverete tutti manuali.

Ho testato la mini console su un TV TCL 55 pollici in modalità gioco e ho scoperto che il controller ha una leggerissima latenza. Inferiore al quarto di secondo e comunque nei limiti ragionevoli per una console plug and play.

Tutti i giochi testati non hanno presentato bug o errori né tantomeno blocchi.

La porta USB sul retro permette di caricare ulteriori titoli attraverso una pendrive formattata in FAT32.

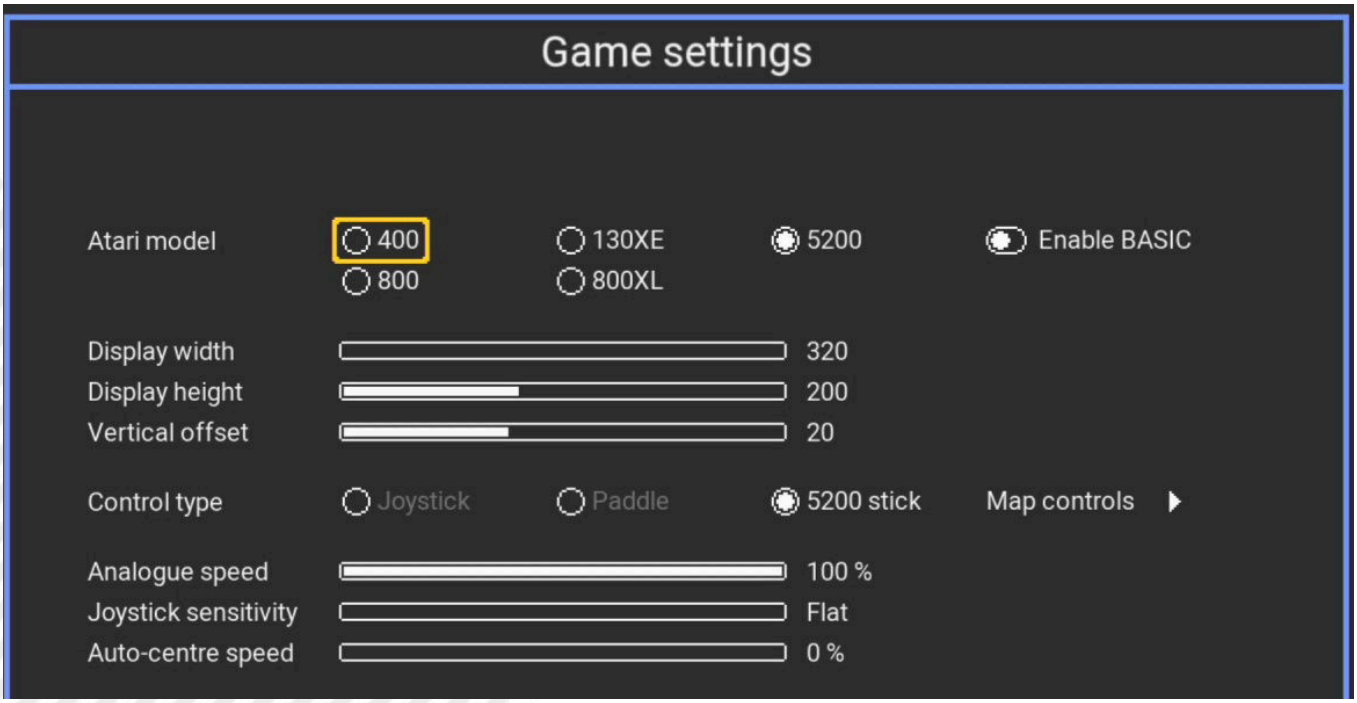


Fig. 5 -Il menù di settaggio per i giochi

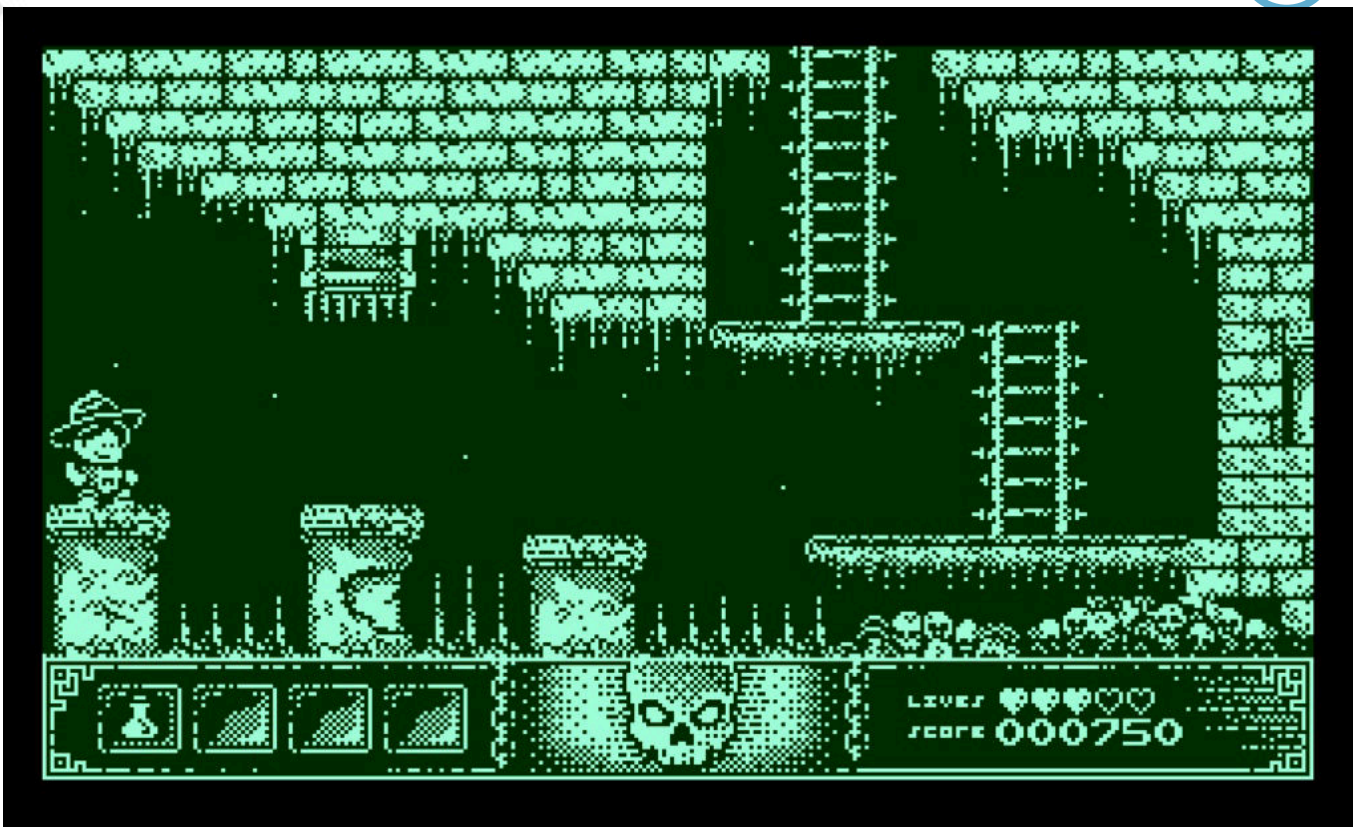


Fig. 6 -Tony Montezuma's Gold gira alla grandissima su THE400 Mini

Questo significa poter accedere a tutta la ludoteca possibile per i sistemi e anche alle nuove meravigliose produzioni. The400 Mini accetta i formati nei file ATR, COM e XEX. Rileva automaticamente il tipo di sistema in cui gira il gioco e se richiede espansioni e accorgimenti di setup.

Premendo MENU' durante il gioco si apre una schermata per configurare il modello ATARI (400, 800, 800XL, 130 XE o 5200, con o senza Basic abilitato), l'altezza e la larghezza del display, l'offset verticale e un modo per rimappare i controlli.

Programmazione, giochi multi-disco e alcuni bug

Parlando di BASIC, si può programmare su THE400 mini formattando una pendrive USB, abilitando il BASIC nelle impostazioni, collegando una tastiera USB in una porta libera e caricando THE400_BASIC.

Questo da accesso al sistema e a un disco per salvare i programmi.

I test con le moderne tastiere sono stati misti. La mancanza di una tastiera "Reale" si fa sentire per impostazioni e comodità. Il settaggio è ovviamente non reale.

La mancanza di una tastiera inclusa è anche il motivo per cui il sistema manca di alcuni titoli importanti, come l'originale Star Raiders e Rescue on Fractalus!. Ma basta un po' di pazienza e si riesce a lavorare.

Il sistema supporta anche i giochi multi-disco con una soluzione ingegnosa: selezioni il primo con il pulsante di fuoco e il secondo con il pulsante S (dorsale) e poi si preme HOME per caricare.

Quando è il momento di inserire il secondo o un terzo disco, si tengono premuti i pulsanti direzionali HOME e GIU'. Sembra macchinosa ma funziona perfettamente.

Il verdetto

Non si può negare che l'Atari 400 mini raggiunga pienamente l'obiettivo: presenta una replica in miniatura di un classico a 8bit, ricco di titoli e impreziosito da diverse funzionalità moderne, da possibilità di aggiungere nuovi titoli e da un buon Joystick di supporto.

Anche i titoli pre installati sono una buona scelta: certo, sarebbe stato bello vedere i classici Nintendo o Pacman e anche il già citato Star Raiders, ma questo problema si supera inserendoli su una pendrive di supporto.

Alcuni dubbi sorgono sul pubblico a cui è destinato; mentre il VCS o le console Nintendo possono fare affidamento su una vasta platea di fanatici, la famiglia Atari 8 bit è meno nota, anche se i suoi giochi sono di una qualità incredibile grazie ad un hardware superiore.

Se si è cresciuti con questo sistema, il the400 mini è il prodotto dei sogni. Emulazione eccellente, possibilità di caricare ogni cosa e aggiornamenti moderni come sistemi di salvataggio e funzione rewind.

VERDETTO: 8/10

Sito Web del prodotto: https://retrogames.biz/

Compilare con Libdragon

di Takahiro Yoshioka – trad Carlo Nithaiah Del Mar Pirazzini

Continuiamo la nostra esplorazione in Libdragon e questa volta proviamo a Compilare.
Passiamo la palla a Takahiro.

Introduzione

La compilazione (o creazione) è il processo che prende il codice sorgente e lo converte in un file binario eseguibile. Ci sono alcuni passaggi intermedi come la preelaborazione e il collegamento, ma per semplicità chiameremo l'intera operazione come "Compilazione". In questa pagina vedremo come iniziare a compilare i progetti Libdragon.

Preparazione della finestra mobile

Il primo passo è far funzionare il tuo motore Docker. Puoi farlo aprendo Docker Desktop.
Se non hai Docker aperto o non hai creato un contenitore Libdragon, potresti ricevere un errore come questo quando esegui Make:

```
libdragon make
```

Questa guida presuppone che tu abbia seguito i passaggi descritti nella sezione Impostazione di Lib.

Esecuzione del comando di compilazione

Il comando di compilazione è abbastanza semplice. Apri semplicemente il tuo terminale/CMD, spostati nella directory corretta ed esegui questo comando:
Ciò richiederà tutto il tuo codice sorgente e produrrà un sacco di file di output inclusa la ROM del tuo Nintendo 64. Se ricevi un errore, *** No rule to make target '/include/n64.mk'. Stop.
Significa che stai utilizzando un file normale make anziché libdragon make.

Spiegazione dei file di output

Una volta compilato il progetto, nella cartella del progetto saranno presenti diversi nuovi file.
Il file principale che ti interesserà è hello.z64 file, che sarà il tuo file ROM N64. Puoi inserirlo nel tuo emulatore preferito ed eseguirlo per vedere come funziona il tuo programma.

Il resto del file viene inserito nella ./build/ directory. Supponendo che tu stia lavorando con il progetto scheletro, sono i seguenti (sarà diverso per progetti più complessi):
• **hello.elf** – Questo è il file in formato collegabile eseguibile. È un file binario che contiene codice oggetto, librerie

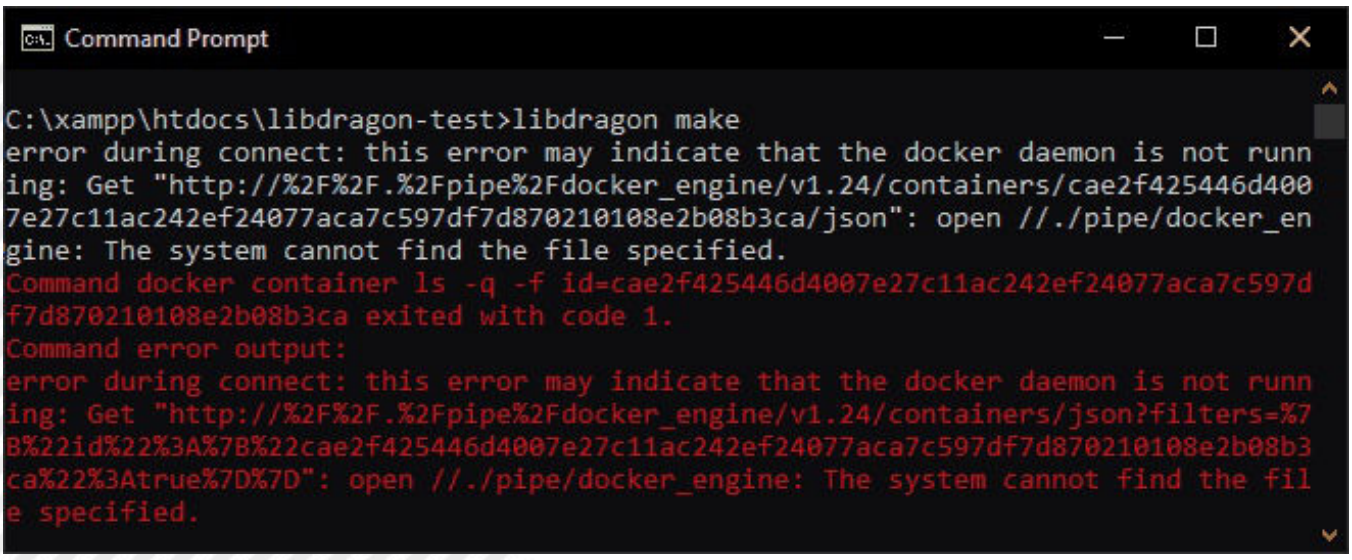


Fig. 1 - Esempio di output del comando: libdragon make

condivise e codice eseguibile.

• **hello.elf.bin** e **hello.elf.sym** – Non sono sicuro di cosa siano, ma sembrano file che accompagnano il file .elf principale che contiene codice macchina binario e informazioni sui simboli.

• **hello.map** – File di testo che contiene informazioni su come è disposta la memoria e sulle varie sezioni di dati/codice dell'eseguibile.

• **main.d** – File di dipendenza. Mostra da quali file dipende la build. In questo semplice programma menzionerà solo main.o da quale è stata creata main.c.

• **main.o** – Il file oggetto per main.c. Contiene il codice compilato utilizzato per l'output nella fase di compilazione della build (prima del collegamento)

Tieni presente che i file chiamati "hello._" si riferiscono alla build nel suo insieme, mentre quelli chiamati "main._" si riferiscono ai singoli file sorgente.

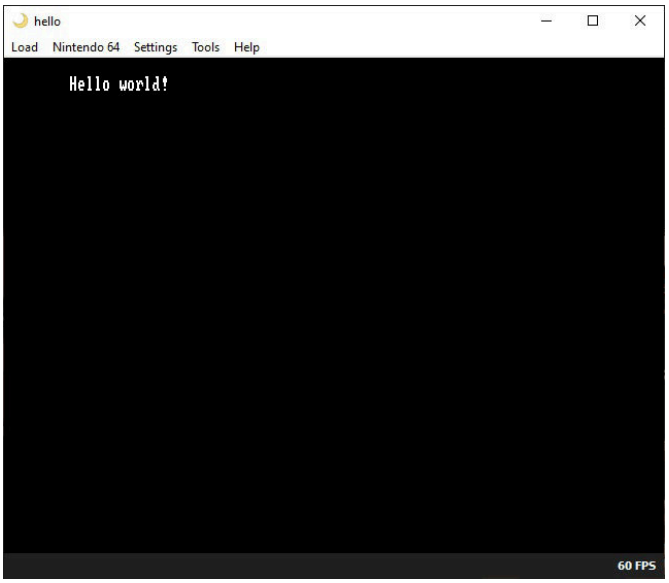
Se utilizzi Git per il monitoraggio delle versioni, potrebbe essere saggio aggiungere questi file all'elenco .gitignore di esclusione poiché cambieranno spesso, occuperanno spazio e non sono molto rilevanti per lo sviluppo del tuo progetto. Possono essere utili per il debug se sai come farlo. Basta aggiungere questo:

```
build/
*.z64
```

Esecuzione della ROM

Esegui la ROM in un emulatore di tua scelta. Tendo a utilizzare Project 64, ma in questo caso utilizzerò Ares.

Dovresti ottenere qualcosa del genere:



Ecco fatto, la tua prima ROM Libdragon N64!

PunyInform - Tutorial 3

di Fredrik Ramsberg - tradotto e adattato per RMW da Gianluca Girelli

Programmazione con PunyInform 1: Le basi

PunyInform è uno strumento per scrivere avventure testuali sia per computer a 8 bit che per computer più recenti. Se avete già installato PunyInform (trovate tutte le informazioni nei numeri precedenti), è ora di iniziare a programmare il vostro primo gioco. Questo tutorial vi guiderà attraverso i primi passi per scrivere un'avventura testuale utilizzando PunyInform.

Il codice sorgente completo del gioco di esempio può essere scaricato dal link a fondo pagina. Inizia quindi qui una serie di articoli tutorial, di cui questo è il primo. Questa prima versione del gioco implementa alcuni ambienti e un oggetto con cui il giocatore può interagire, ma non c'è modo di vincere o perdere: ciò verrà trattato nel prossimo tutorial.

Iniziamo a scrivere un nuovo gioco - Fig. 1

Direttive del compilatore

Le prime due righe del codice sono le direttive del compilatore, che aiutano a rendere il file di gioco il più piccolo possibile. In genere si desidera avere queste righe nella parte superiore del codice sorgente di ogni gioco PunyInform che scriviamo.

Commenti

Nelle righe 3, 4, 9, 16 e 20 possiamo vedere come inserire commenti per rendere il codice più leggibile: un punto esclamativo significa che il resto della riga è un commento a testo libero (a meno che il punto esclamativo non sia seguito da % e la riga appaia all'inizio del file; in tal caso è trattata come una direttiva del compilatore.)

Storia e titolo

Il secondo passo è dichiarare il nome del gioco/storia e il titolo, qualcosa di simile a uno slogan. Dobbiamo sempre

```
1  !% -~S
2  !% $OMIT_UNUSED_ROUTINES=1
3  !% $ZCODE_LESS_DICT_DATA=1
4
5  Constant Story      "The Mansion at the Edge of Town";
6  Constant Headline   "^An interactive nightmare.^";
7
8  ! Constants to control PunyInform are defined here
9  Constant DEBUG;
10 Constant OPTIONAL_FULL_DIRECTIONS;
11 Constant INITIAL_LOCATION_VALUE = Entrance;
12
13 Include "globals.h";
14
15 ! PunyInform extensions and entry point routines are placed here
16
17 Include "puny.h";
18
19 ! Verb grammar and actions go here
20
```

Fig. 1 - Iniziamo a scrivere un nuovo gioco

definire lo "story name" e, nella maggior parte dei casi, vorremo definire anche un titolo. Tienete presente che ^ può essere utilizzato nelle stringhe per indicare caratteri di nuova riga. (Inoltre, ~ può essere usato per indicare i caratteri delle virgolette doppie, ma non è usato in questo codice.)

Definizione delle costanti

Nella sezione successiva vengono definite le costanti che controllano PunyInform. Maggiori informazioni sulle costanti possono essere reperite nel manuale incluso nella cartella di ogni release di PunyInform, alla voce Customizing the Library. Per alcune costanti, come INITIAL_LOCATION_VALUE, è necessario fornire un valore. Per le altre costanti, tuttavia, è sufficiente definire semplicemente la costante (dandole effettivamente il valore 0). La libreria PunyInform controlla se la costante è stata definita o meno e modifica di conseguenza alcuni aspetti del suo comportamento. Constant DEBUG; viene utilizzato per abilitare la modalità debug, il che significa che sarà disponibile una serie di verbi di debug, quindi puoi digitare cose come "GONEAR TEDDY BEAR" per teletrasportarti dove si trova l'orsacchiotto, o "ACTIONS" per vedere quali azioni vengono attivate da ciò che digiti sulla tastiera. È molto utile averlo durante lo sviluppo e il test del gioco, ma ricordate di disattivarlo prima di rilasciare il gioco al pubblico. Constant OPTIONAL_FULL_DIRECTIONS; invece abilita le direzioni nord-ovest, nord-est, sud-ovest e sud-est. Se vuole usare direzioni multiple (invece delle quattro standard nord, sud, est e ovest), dovete definire questa costante. Ricordate che però rende il gioco un pochino più grande e un pochino più lento.

Costant INITIAL_LOCATION_VALUE = Entrance; dice al gioco da dove il giocatoreinizierà la sua avventura. Tenete inoltre presente che ogni istruzione o affermazione deve terminare con un punto e virgola.

Includere globals.h

Dopo aver definito tutte le costanti di cui abbiamo bisogno, è il momento di includere il file “globals.h”. Questo imposta le variabili globali, le costanti, ecc. necessarie alla libreria PunyInform.

Estensioni della libreria e entry point routines

E' il momento di includere tutte le estensioni della libreria che vogliamo utilizzare (confronta Extensions nel manuale per saperne di più) e scrivere tutte le entry point routines di cui abbiamo bisogno (Non c'è bisogno di leggerlo ora, ma quelle supportate sono elencate sotto Entry Point Routines nel manuale PunyInform e ciò che fanno è descritto nel Manuale di The Inform Designer sotto Entry Point Routines.) [NDR: la filosofia di gioco è basata sul manuale ufficiale del linguaggio Inform6, "The Inform Designer's Manual" scritto da Graham Nelson, giunto alla versione 4. Trovate il link a fondo pagina.]. In questo particolare gioco, abbiamo deciso di non utilizzare alcuna estensione o entry point routines. Possiamo sempre cambiare idea in seguito, in caso di necessità.

Includere puny.h

A questo punto includiamo il secondo ed ultimo file della libreria, chiamato “puny.h”. Questo file, a sua volta, includerà tutti gli altri file della libreria.

Verbi, grammatica e azioni

Infine, definiamo nuovi verbi, estendiamo quelli esistenti con una nuova grammatica e definiamo nuove azioni. Lasciamo questa sezione vuota per ora.

Oggetti di gioco - Fig.2

Ora è il momento di definire gli oggetti del gioco. In genere si crea un oggetto per ogni locazione del gioco e un oggetto per ogni cosa con cui il giocatore può interagire. Una cosa può essere qualcosa di tangibile come una chiave o un'auto, ma può anche essere qualcosa come l'aria, l'acqua, la costa vista da un aereo, o anche un odore o un suono. Gli oggetti possono essere utilizzati anche per concetti astratti e concetti di gioco come pensieri, conversazioni, menu, ecc. Gli oggetti hanno una gerarchia, in cui un oggetto può essere "dentro" un altro oggetto. Il significato di "in" dipende dalle proprietà dell'oggetto contenitore (il cosiddetto genitore) e dell'oggetto contenuto (il cosiddetto figlio). Essere in un altro oggetto di solito significa che il figlio è nel genitore, sul genitore, tenuto dal genitore, indossato dal genitore o parte del genitore.

Stanze

Una stanza (anche detta locazione) è un oggetto che non

```
! The game objects begin here
Object Entrance "Entrance"
    with
        name 'front' 'door' 'cubicle' 'stairs',
        description "You are in the entrance area of the mansion. A wide set of
            stairs lead up. There is a small cubicle in the southeast corner,
            while the front door is to the south.",
        u_to UpstairsHallway,
        se_to Cubicle,
        in_to Cubicle,
        s_to "The front door has apparently been nailed shut.",
        cant_go "There's nothing of interest in that direction.",
    has light;

Object Cubicle "Cubicle"
    with
        initial "You squeeze in through the small opening, hitting your head
            against the doorway in the process.^",
        description "It's quite cosy in here. There are some marks on the wall
            where a desk probably used to stand. The exit is to the north.",
        out_to Entrance,
        n_to Entrance,
    has light;

```

Fig. 2 - Oggetti di gioco

ha un genitore e in cui il giocatore può trovarsi. Iniziamo definendo la nostra prima stanza. Per prima cosa digitiamo Object seguito dal nome interno dell'oggetto, in questo caso scegliamo di chiamarlo Entrance. Poi arriva la stringa del nome dell'oggetto. Questo è il nome che verrà stampato a video per questo oggetto nel gioco, racchiuso tra virgolette doppie. Per le stanze, è pratica comune scrivere in maiuscolo il nome stampato. Noterete che il nome interno e il nome stampato di un oggetto sono spesso identici o molto simili. Tuttavia, il nome interno può essere composto solo dalle lettere A-Z e a-z, cifre e trattino basso, senza spazi o altri caratteri speciali. E mentre molte località possono condividere lo stesso nome stampato (ad esempio una serie di locazioni potrebbero avere lo stesso nome stampato di "Foresta" o "Brughiera"), il nome interno di ciascuna località deve essere univoco. Abbiamo scelto di mettere Object Entrance "Entrance" sulla stessa riga e poi andare a capo prima dell'istruzione with , ma è solo una scelta stilistica: si possono avere interruzioni di riga ovunque si desideri nella definizione di un oggetto.

Proprietà

Successivamente vengono visualizzate le proprietà dell'oggetto e la parola chiave with segna l'inizio di questa sezione. Una proprietà può contenere uno o più valori a 16 bit. Ogni valore viene spesso utilizzato per memorizzare un numero, un ID oggetto, l'indirizzo di una parola del dizionario, l'indirizzo di stringa o l'indirizzo di una routine. Se lo si usa come numero, può contenere qualsiasi numero intero compreso tra -32768 e +32767. Dopo l'ultimo valore di una proprietà, si deve digitare una virgola per dire che questa proprietà finisce qui. Tenete presente che un oggetto ottiene solo le proprietà che per esso vengono definite. Ciò significa che se volete creare un'uscita a nord in una stanza più avanti nel gioco, ma non dovrebbe essercene una dall'inizio, dovete includere n_to 0, nella definizione dell'oggetto. Ciò garantisce che n_to esista e abbia un valore, in modo che il valore possa essere successivamente modificato. Per prima cosa diamo all'oggetto Entrance la proprietà name e gli assegniamo quattro valori, ciascuno (l'indirizzo di) una parola del dizionario. Quando diamo a una stanza la proprietà name,

significa che se il giocatore usa una di queste parole in questa stanza, il gioco dovrebbe stampare "You don't need to use that word in this game." o qualcosa del genere. Questo può essere usato per aiutare il giocatore a concentrarsi su ciò che è importante nel gioco. Ricordate che le parole del dizionario sono sempre racchiuse tra virgolette singole. Se una parola del dizionario è composta da un solo carattere, è necessario aggiungere // dopo quel carattere, per distinguerlo da un carattere, che utilizza anch'esso virgolette singole. Quindi: 'n' è un carattere, mentre 'n//' è una parola del dizionario. Assegnamo ora la proprietà description, obbligatorio se l'oggetto è una stanza. La descrizione può essere una stringa o una routine che stampa del testo, e questo viene stampato o eseguito quando il giocatore entra nella stanza o digita LOOK. Come potete vedere, una stringa può contenere caratteri di fine riga e quindi la riga successiva può iniziare con tabulazioni o spazi: tutto questo viene contratto dal compilatore in un singolo carattere di spazio. Successivamente, scegliamo di specificare le uscite da questa stanza. Per questo utilizziamo le proprietà n_to, s_to, e_to, w_to, u_to, d_to, in_to e out_to . Se abbiamo definito costant OPTIONAL_FULL_DIRECTIONS , possiamo anche usare ne_to, nw_to, se_to e sw_to . Ogni uscita può avere un singolo valore (non una lista di valori), e questo valore solitamente è (A) una stanza o (B) una stringa da stampare se il giocatore tenta di andare in quella direzione. Definiamo quindi cant_go , che è un messaggio da stampare se il giocatore tenta di andare in una direzione per la quale non abbiamo specificato un valore. Se non definiamo la proprietà cant_go, verrà visualizzato il messaggio standard "You can't go that way.".

Attributi

Esaminiamo ora la parola chiave has , il che significa che definiremo gli attributi di questo oggetto. Un attributo è un flag a un bit, che può essere solo vero o falso. A differenza delle proprietà, tutti gli attributi esistono per tutti gli oggetti, quindi anche se non si include un attributo nella definizione dell'oggetto, si può comunque modificarlo più avanti nel gioco. In questo caso diamo alla stanza l'attributo light, nel senso che è la stanza stessa a fornire

```
Object UpstairsHallway "Upstairs Hallway"
    with
        description "Nothing much to see here really. There are some doors, but
            they have all been nailed shut. The stairs lead down to the west.",
        d_to Entrance,
        w_to Entrance,
    has light;

Object -> Painting "painting"
    with
        name 'painting',
        initial "Hanging on the west east wall is a dark oil painting.",
        description "It's a lovely old painting of a crying clown.",
        after [;
            Examine:
                give self general;
        ],
        short_name [;
            if(self has general) {
                print "painting of a clown";
                rtrue;
            }
        ];

```

Fig. 3 - oggetto chiamato UpstairsHallway

luce. Ciò significa che il giocatore può vedere nella stanza (nel senso che può vedere cio' che contiene ed intaragire con eventuali oggetti), senza portare la propria lampada. Successivamente inseriamo un punto e virgola, che segna la fine della definizione dell'oggetto.

L'oggetto Cubicle

Questo oggetto porta solo una cosa nuova sul tavolo: la proprietà initial. Se utilizzata in una stanza, questa proprietà viene stampata subito prima del nome della stanza quando il giocatore entra nella stanza. Ecco come appare:

```
Cubicle                                    Score: 0  Moves: 1

The Mansion at the Edge of Town
An interactive nightmare.
Release 1 / Serial number 230918 / Inform v6.41 PunyInform v5.0 DR

Entrance
You are in the entrance area of the mansion. A wide set of stairs lead
up. There is a small cubicle in the southeast corner, while the front
door is to the south.

> se
You squeeze in through the small opening, hitting your head against
the doorway in the process.

Cubicle
It's quite cosy in here. There are some marks on the wall where a
desk probably used to stand. The exit is to the north.

>|
```

Piano superiori e oggetti normali (non stanze)

Ricordate che abbiamo scritto nell'oggetto Entrance che salire (u_to) porterebbe a UpstairsHallway? Ciò significa che dobbiamo definire un oggetto chiamato UpstairsHallway da qualche parte nel codice. Si veda la Fig. 3

Questa location non illustra alcun concetto nuovo per noi, ma mostra come a volte sia conveniente avere due uscite diverse che conducono alla stessa posizione. Con il testo che dice che le scale scendono a ovest, ha senso avere sia d_to che w_to che conducono alla stanza in fondo alle scale. Ora ecco che arriva il nostro primo oggetto che non è una stanza. Stiamo parlando di un dipinto. Mettendo una freccia (->) dopo Oggetto diciamo al linguaggio che questo oggetto è all'interno dell'ultimo oggetto che non aveva una freccia, quindi il dipinto è nella stanza UpstairsHallway. Poi vengono le properties. Ricordate che la proprietà description è obbligatoria per le stanze? Non è così per gli oggetti normali. In effetti, non esistono proprietà obbligatorie per gli oggetti regolari. In genere, vogliamo che il giocatore sia in grado di fare riferimento agli oggetti che incontra (non alle stanze, ma praticamente a tutti gli altri oggetti). Il modo più semplice per abilitare ciò è dare all'oggetto una proprietà name e lasciare che contenga un elenco di parole del dizionario che sono sostantivi e forse aggettivi che possono essere usati per riferirsi a questo oggetto. In questo caso, ci accontentiamo semplicemente di 'painting'. E ricordate di racchiudere sempre le parole del dizionario tra singoli apici. Poi arriva la proprietà initial. Per gli oggetti normali viene utilizzata per descrivere l'oggetto finché il giocatore non lo ha raccolto, quindi possiamo usarla per descrivere il dipinto in un modo che abbia senso solo nel luogo in cui si trova il dipinto all'inizio del gioco. Poi arriva la proprietà description. Per gli oggetti normali, questa viene stampata se il giocatore esamina gli oggetti. Può essere una stringa o una routine che stampa un testo. Successivamente, definiamo la proprietà after. Questa è una routine (inizia con [; e termina con]) che viene chiamata dopo che un'azione è riuscita. In questa routine, si digita il nome di un'azione che il giocatore può eseguire dove questo è l'oggetto principale (chiamato anche noun), i due punti e il codice che si desidera eseguire se questa azione è stata intrapresa. Tutte le azioni possibili sono elencate nel manuale sotto actions e mentre si gioca in modalità DEBUG tutte le azioni eseguite dal linguaggio possono essere visualizzate dopo aver digitato il comando ACTIONS. In questo caso vogliamo eseguire del codice dopo che il giocatore ha esaminato il dipinto, e questo viene reso possibile dall'istruzione give self general; Analizziamolo. give è un'istruzione per impostare un attributo per un oggetto. self è l'oggetto su cui viene eseguito questo codice (quindi in questo caso avremmo potuto digitare Painting e ottenere lo stesso effetto). general è un attributo multiuso. Decidiamo noi cosa significa per ogni oggetto. In questo caso abbiamo deciso che per il dipinto significhi “Il giocatore ha esaminato il dipinto?”. L'attributo è azzerato all'inizio del gioco e non appena il giocatore esamina il dipinto viene impostato a "general". Alla fine di una regola after, come questa, di solito possiamo scegliere di restituire true o false, dove true significa "Abbiamo stampato un messaggio adatto, quindi la routine di azione non dovrebbe stamparne uno.". Tuttavia, Examine è un caso speciale. Dato che tutto ciò che fa è stampare qualcosa, stampa il suo messaggio prima che venga chiamata la routine after, e non importa

cosa restituiamo. Nota: le routine nelle proprietà restituiscono false alla fine, quindi se non restituisci nulla in modo esplicito, restituiscono false. Le routine con nome ("named routines" vedi più avanti) restituiscono true alla fine. Esaminiamo ora short_name . Questa è un'opportunità per sovrascrivere il nome fornito tra virgolette doppie sulla prima riga di questo oggetto. Può essere una stringa o una routine. Se è una routine, viene richiamata la routine e se la routine restituisce true si presume che sia stato stampato il nome breve completo dell'oggetto e non viene stampato altro. Se restituisce false, viene stampato il nome normale dell'oggetto. Qui controlliamo se il dipinto ha l'attributo general settato (il che significa che il giocatore ha esaminato l'oggetto), e se lo ha, stampiamo la stringa "dipinto di un clown" e restituiamo true. In caso contrario, arriviamo alla fine della routine, il che significa che la routine restituisce false e viene invece stampato il nome normale dell'oggetto. Nota: true è solo un sinonimo di 1 e false è un sinonimo di 0. Si può restituire qualsiasi valore utilizzando l'istruzione return, ma esistono forme abbreviate per quando si vuole invece restituire true o false. return true; and return 1; e rtrue; sono quindi equivalenti. Proprio come return false; return 0; e rfalse; sono equivalenti. Nota: un'istruzione if può eseguire una singola istruzione (if(x > 5) print "X è grande!";) o più istruzioni racchiuse tra parentesi graffe (if(x > 5) { print "X è grande!"; y = x; }). I ritorni a capo e le tabulazioni o gli spazi aggiuntivi vengono aggiunti solo per motivi di leggibilità e non influiscono sull'esecuzione del programma. Vediamo questo codice in azione.

```
Upstairs Hallway                                Score: 0  Moves: 5

> gonear painting
Upstairs Hallway
Nothing much to see here really. There are some doors, but they have
all been nailed shut. The stairs lead down to the west.

Hanging on the west east wall is a dark oil painting.

> get painting
Taken.

> i
You're carrying a painting.

> examine the painting
It's a lovely old painting of a crying clown.

> i
You're carrying a painting of a clown.

>|
```

Questi sono tutti gli oggetti del gioco, almeno per questa prima iniziale.

La routine Initialise

È rimasta solo una piccola routine obbligatoria. Ogni gioco PunyInform deve fornire una routine chiamata Initialise. Si veda la Fig. 4

Scrivere una "named routine" significa semplicemente iniziare con [quindi digitare il nome della routine, eventuali variabili locali necessarie e un punto e virgola. Quindi si scrive il codice e si finisce con];
Initialise viene richiamata dalla libreria all'avvio del gioco. In questo caso, l'unico codice di cui abbiamo bisogno qui è l'istruzione print per stampare il testo iniziale all'avvio del gioco.

Di seguito il codice completo di questo gioco "minimale", che verrà affinato e espanso negli articoli successivi. Buon divertimento!

```
!% -~S
!% $OMIT_UNUSED_ROUTINES=1
!% $ZCODE_LESS_DICT_DATA=1

Constant Story      "The Mansion at the Edge
of Town";
Constant Headline   "^An interactive
nightmare.^";

! Constants to control PunyInform defined here
Constant DEBUG;
Constant OPTIONAL_FULL_DIRECTIONS;
Constant INITIAL_LOCATION_VALUE = Entrance;

Include "globals.h";

! PunyInform extensions and entry point
routines are placed here

Include "puny.h";

! Verb grammar and actions go here

! The game objects begin here
Object Entrance "Entrance"
  with
      name 'front' 'door' 'cubicle' 'stairs',
      description "You are in the entrance
```

```
69  ! And finally, the Initialise routine goes here
70  [Initialise;
71      print "^^You and your friends went out to the old abandoned mansion on the
72          outskirts of town. Once in the house, you remember getting separated
73          from the group, and then all went black. As you come to, several hours
74          have passed, you seem to be alone, and everything is quiet.^^";
75  ];
76
```

Fig. 4 - La routine Initialise

```
area of the mansion. A wide set of
        stairs lead up. There is a small
cubicle in the southeast corner,
       while the front door is to the south.",
    u_to UpstairsHallway,
    se_to Cubicle,
    in_to Cubicle,
    s_to "The front door has apparently
been nailed shut.",
    cant_go "There's nothing of interest
in that direction.",
  has light;

Object Cubicle "Cubicle"
  with
    initial "You squeeze in through the
small opening, hitting your head
        against the doorway in the process.^",
    description "It's quite cosy in here.
There are some marks on the wall
        where a desk probably used to stand.
The exit is to the north.",
    out_to Entrance,
    n_to Entrance,
  has light;

Object UpstairsHallway "Upstairs Hallway"
  with
    description "Nothing much to see here
really. There are some doors, but
        they have all been nailed shut. The
stairs lead down to the west.",
    d_to Entrance,
    w_to Entrance,
  has light;

Object -> Painting "painting"
  with
    name 'painting',
    initial "Hanging on the west east wall
is a dark oil painting.",
    description "It's a lovely old painting
of a crying clown.",
    after [;
        Examine:
        give self general;
    ],
    short_name [;
        if(self has general) {
            print "painting of a clown";
            rtrue;
        }
    ];

! And finally, the Initialise routine goes here
[Initialise;
  print "^^You and your friends went out to
the old abandoned mansion on the
    outskirts of town. Once in the house,
you remember getting separated
    from the group, and then all went black.
As you come to, several hours
    have passed, you seem to be alone, and
everything is quiet.^^";
];
```

LINK UTILI

- https://drive.google.com/drive/folders/1iYRoJYkPXx7RYnd-tl0gT74L2Sa04mSp
- https://www.inform-fiction.org/manual/about_dm4.html

Da numero a cifre (per C64 – per principianti)

di Eugenio Rapella

Supponiamo di avere un numero naturale X formato da quattro cifre (in base 10). Per qualche motivo (ad esempio per risolvere un "Quesito con la Susi"), abbiamo bisogno di ottenere separatamente le quattro cifre.

In definitiva, partendo da X, vogliamo ottenere le quattro variabili U, D, C, M che contengano rispettivamente la cifra delle unità, quella delle decine, quella della centinaia e quella delle migliaia (così, se X=7235, avremo U=5, D=3, C=2 e M=7).

Ecco una possibilità per il nostro C64 (PROG1):

```
10 v1=int(x/10):u=x-10*v1
20 v2=int(v1/10):d=v1-10*v2
30 m=int(v2/10):c=v2-10*m
```

Per capire come funziona il programmino è sufficiente seguire un esempio. Se X=7235, abbiamo X/10 = 723.5 e V1=INT(723.5)=723.
Dunque U=7235-10*723=5 come desiderato. Ora V1/10=72.3 quindi V2=72 e D=723-10*72=3. Infine V2/10=7.2 quindi M=7 e C=72-10*7=2.

Un'altra possibilità è quella di trasformare X in una stringa e sfruttare la funzione MID$(..) per estrarre una singola lettera dalla stringa. Alla fine, otterremo il valore numerico con la funzione VAL(..).

Ecco il PROG2:

```
10 x$=str$(x)
20 u=val(right$(x$,1))
30 d=val(mid$(x$,4,1))
40 c=val(mid$(x$,3,1))
50 m=val(mid$(x$,2,1))
```

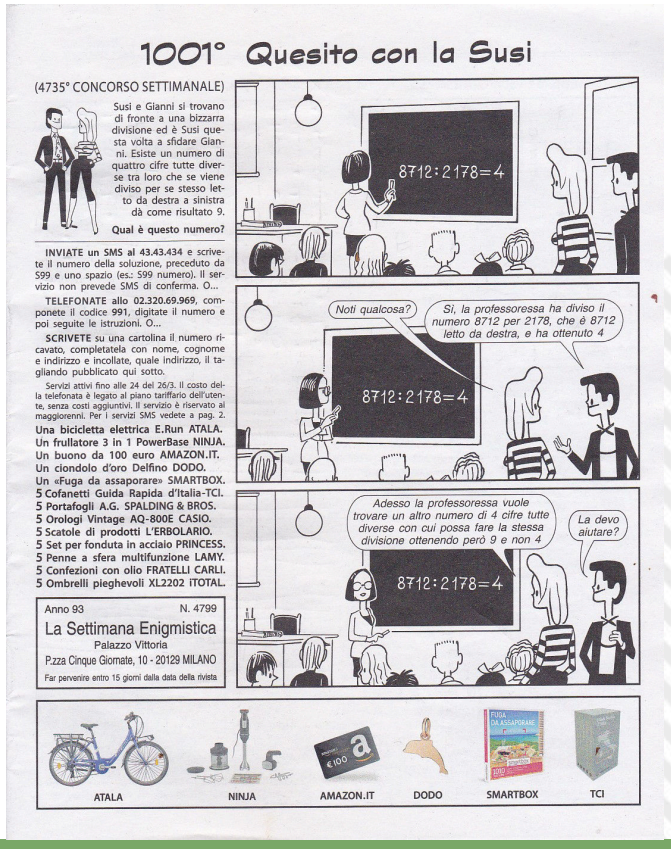
Da notare che la funzione STR$(..) trasforma il valore numerico in stringa anteponendo uno spazio. Così, se X=7235, avremo una stringa di cinque caratteri: X$=" 7235".

RIGHT$(X$,1) restituisce l'ultimo carattere della stringa ("5") per cui U=VAL(RIGHT$(X$,1)) diviene il valore numerico 5.
Poiché la funzione MID$(A$,H,K) restituisce una stringa formata da K caratteri della stringa A$ a partire dalla posizione H (così, se A$="MAGAZINE", avremo MID$(A$, 3,4)="GAZI"), D sarà 3, C assumerà il valore 2 e M il valore 7.

Un confronto tra PROG1 e PROG2, in termini di velocità, non ha evidenziato grandi differenze; i due programmi possono essere facilmente modificati se il numero delle cifre è diverso da quattro.

Ed eccoci ad un nuovo "Quesito con la Susi", sempre da "La Settimana Enigmistica":

L'insegnante mostra che, se dividiamo 8712 per 2178, che è il primo numero scritto "all'indietro", si ottiene esattamente 4. La richiesta è quella di trovare un numero di quattro cifre che, operando in maniera analoga, dia esattamente 9 anziché 4.



Ecco come procede il nostro C64 (PROG3):

```
10 for k=1000 to 9999
20 v1=int(k/10):u=k-10*v1
30 v2=int(v1/10):d=v1-10*v2
40 m=int(v2/10):c=v2-10*m
50 h=1000*u+100*d+10*c+m
60 if 9*h=k then print k,h
70 next
```

Per ogni K che va da 1000 a 9999, vengono estratte da K le singole cifre e, alla 50, viene costruito il numero H che è K … scritto all'indietro.
Alla 60 si controlla se K/H è esattamente uguale a nove nel qual caso K e H vengono stampati.

Un'alternativa al PROG3, è il seguente PROG4 dove, a partire dalle singole cifre, si costruiscono i due numeri K e H. Si noti che la cifra delle migliaia, M, parte da uno in modo che K sia, in effetti, un numero di quattro cifre:

```
10 for m=1 to 9:for c=0 to 9
20 for d=0 to 9:for u=0 to 9
30 k=1000*m+100*c+10*d+u
40 h=1000*u+100*d+10*c+m
50 if 9*h=k then print k,h
60 next:next:next:next
```

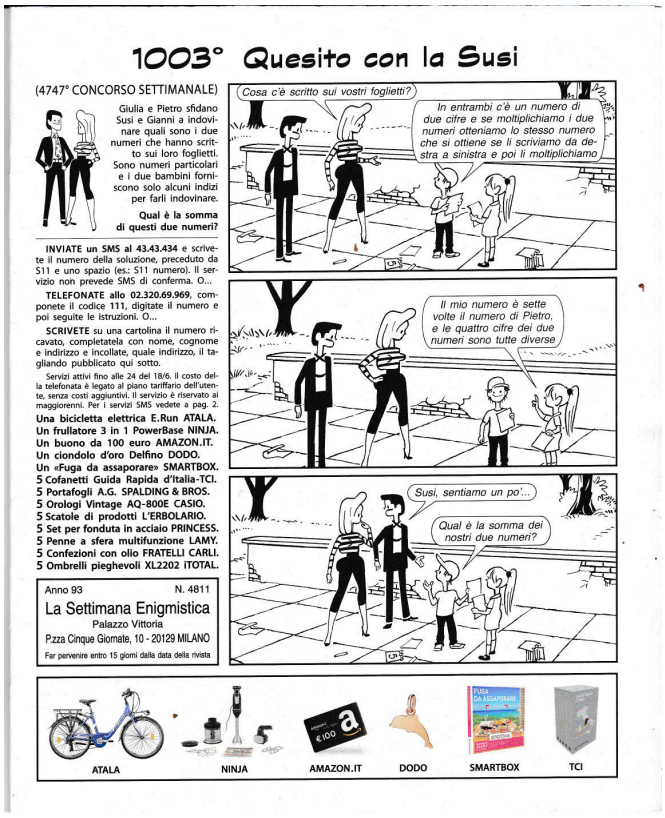
Entrambi i programmi passano in rassegna molti più candidati di quanto sia necessario; ovviamente, non ci si aspetta che i lettori de "La Settimana Enigmistica" risolvano il problema con l'aiuto del C64 e qualche scorciatoia ci deve pur essere (al solito, il mio Commodore mi vieta di "spoilerare" i suoi programmi dando suggerimenti).

I nostri PROG3 e PROG4 hanno però il vantaggio di fornire rapidamente altre informazioni:

- Quando viene trovata una soluzione, l'elaborazione prosegue comunque. Alla fine si ha una sola stampa, a conferma che la soluzione è unica.
- Nel testo originale del quesito viene segnalato che le cifre devono essere diverse tra loro. I programmi ignorano questa richiesta: la soluzione è comunque unica anche senza questa limitazione.
- Se l'istruzione IF 9*H=K THEN PRINT K,H viene modificata in IF 4*H=K THEN PRINT K,H si ritrova solo la coppia di numeri 8712, 2178. Anche nell'esempio proposto all'inizio, la soluzione è unica.
- Se in luogo di 9 o 4, eseguo il programma con uno dei valori 2,3,5,6,7,8 non si trova alcuna soluzione.
- Naturalmente con IF H=K THEN PRINT K,H si ottengono tutti i numeri di quattro cifre … "palindromi": 1001, 1111, 1221, … ,9889, 9999.

Mentre stavo scrivendo questo articolo, su un nuovo numero de "La Settimana Enigmistica" è comparso un nuovo "Quesito con la Susi":

Susi chiacchiera con Giulia e Pietro.
Susi: "Cosa c'è scritto sui vostri foglietti?"
Pietro: "In entrambi c'è un numero di due cifre e, se moltiplichiamo i due numeri, otteniamo lo stesso numero che si ottiene se li scriviamo da destra a sinistra e poi li moltiplichiamo".
Giulia: "Il mio numero è sette volte il numero di Pietro e le quattro cifre dei due numeri sono tutte diverse".
Pietro: "Qual è la somma dei nostri due numeri?"



Indichiamo con X1 il numero sul foglietto di Pietro. Se A

rappresenta la cifra delle decine e B quella delle unità, avremo X1=10*A+B. Analogamente, se Y1 rappresenta il numero sul foglietto di Giulia, con C cifra delle decine e D cifra delle unità, avremo Y1=10*C+D.

Il PROG5 lavora seguendo il testo in maniera molto letterale:

```
100 for a=1 to 9:for b=1 to 9
110 for c=1 to 9:for d=1 to 9
120 if a=b or a=c or a=d then 220
130 if b=c or b=d then 220
140 if c=d then 220
150 x1=10*a+b:y1=10*c+d
160 if y1<>7*x1 then 220
170 x2=10*b+a:y2=10*d+c
180 if x1*y1<>x2*y2 then 220
190 s=s+1:print"sol n. ";s
200 print"x1=";x1;"y1=";y1
210 print"prodotto:";x1*y1
220 next:next:next:next
```

Alla 100-110 si aprono i quattro cicli FOR-NEXT: le cifre variano da 1 a 9; lo zero è escluso in modo che tutti e quattro i numeri in gioco risultino di due cifre.

Le quattro cifre devono essere tutti differenti; quando questo non avviene, una delle tre IF (120-140) rimanda alla scelta dei prossimi candidati.

Alla 150 si calcola il possibile valore di X e quello di Y; se Y non è sette volte X, la coppia non può essere la soluzione (istr. 160).

Alla 170 si calcolano i valori di X e Y scritti "all'indietro" (X2,Y2): se il prodotto di questi non coincide col prodotto dei numeri originali, la coppia (X1,Y1) non è la soluzione del quesito (istr. 180).

Alla 190 si arriva solo se tutte le richieste sono soddisfatte; il programma trova subito una soluzione, ma continua la ricerca.

Dopo circa cinque minuti, l'emulatore VICE (no Warp Mode), termina l'elaborazione senza ulteriori stampe: la soluzione è unica.

Per la verità, se il prodotto dei due numeri originali, 10*A+B e 10*C+D, deve coincidere con il prodotto dei due scritti "all'indietro", deve aversi:
(10*A+B)*(10*C+D)=(10*B+A)*(10*D+C).
Eseguendo qualche passaggio, si deduce 99*A*C=99*B*D ovvero A*C=B*D.

Se volete, potete sostituire l'istruzione 170 con 170 IF A*C<> B*D THEN 220 ed eliminare l'istruzione 180: il programma sarà un pochino più corto e un pochino più veloce.

Naturalmente, anche questo "Quesito con la Susi" si può risolvere senza scomodare il nostro Commodore (il numero sul foglietto di Giulia non può essere inferiore a 12 – il più piccolo numero di due cifre con cifre diverse -, ma non può superare 14 – in quanto 14*7=98 è il massimo valore possibile sul foglietto di Pietro, dunque …), ma il nostro PROG5, opportunamente modificato, può essere utile per rispondere ad altre domande.

Ad esempio, quante sono le coppie di numeri di due cifre, tutte diverse, il cui prodotto è identico a quello che si ottiene moltiplicando i numeri scritti all'indietro? Non sono poche, eccone alcune: 12*63=21*36; 13*62=31*26; 14*82=41*28, 23*64=32*46, … , 36*84=63*48.

Grandi fattoriali (per C64 – per principianti)

di Eugenio Rapella

Il nostro C64 non ha bisogno di un gran programma per calcolare il fattoriale di un numero naturale "n" (si indica con "n!" ed è definito come il prodotto di tutti gli interi tra 1 e "n", con la convenzione di porre 0!=1; così 4! =4*3*2*1=24; i fattoriali entrano in gioco in vari capitoli della matematica come il Calcolo Combinatorio, le serie ecc.), per la verità se la cava con una sola istruzione:

```
1 input" n = ";n:f=1:for k=1 to
n:f=f*k:next:print n;"! = ";f
```

Se al RUN chiediamo il calcolo di 10!, otteniamo subito 3628800 e, fino a 12, otteniamo i valori esatti: 12! =479001600. Per 13! il C64 passa alla notazione esponenziale: 13!=6.2270208E+09 ovvero 6,227*10^9 è il valore approssimato di 13 fattoriale. Si va avanti così fino a 33! = 8.68331762E+36. Dal 34 in poi il nostro amato Commodore getta la spugna e segnala un ?OVERFLOW ERROR IN 1.

In internet si trovano numerosi siti che forniscono il valore esatto di "n!" anche per n>33. Al C64 la cosa non va a genio: "Se ci riescono altri, posso riuscirci anch'io!". Vediamo come se la cava.

Al crescere di "n", "n!" cresce molto rapidamente; per prima cosa chiediamo al nostro Commodore di valutare da quante cifre sarà formato "n!" (va da sé che stiamo lavorando in base 10).
Abbiamo bisogno di rinfrescare alcune nozioni relative ai logaritmi; utilizzeremo i logaritmi in base 10. Il logaritmo decimale di un numero naturale k>0 è strettamente legato al numero di cifre che lo formano. Se, ad esempio, k=7203, avremo 1000 ≤ k < 10000. Poiché il logaritmo è una funzione crescente, si avrà 3=log(1000)≤log(k)<log(10000)=4, dunque log(k)=3, …. . In generale, se k≥1, il numero delle cifre che formano "k" è int(log(k))+1 (nel nostro esempio, log(k)= 3,…. da cui int(log(k))=3 e int(log(k))+1=4 che è proprio il numero di cifre di 7203)

Poiché "n!" è il prodotto degli interi da 1 a "n", log(n!) = log(1)+log(2)+ … + log(n) (… ricordate? "Il logaritmo del prodotto è la somma dei logaritmi…"). La funzione LOG del C64 fornisce i logaritmi in base "e" (logaritmi "naturali"), ma è facile risalire al logaritmo in base 10 (logaritmo "decimale") utilizzando la formula del cambiamento di base: LOG10(X)=LOG(X)/LOG(10).
Ecco allora il miniprogramma che ci consente di determinare da quante cifre è formato "n!" :

```
10 input" n = ";n
20 for h=1 to n:s=s+log(h):next
30 print" il numero di cifre di n! e' ";int(s/
log(10))+1
```

Scopriamo allora che 100! è un numero formato da ben 158 cifre, 500! da 1135 cifre e 1000! raggiunge le 2568 cifre!

Per il calcolo effettivo di questi numeroni torniamo al programma iniziale. Mentre il contatore K si mantiene moderato, è la variabile F che cresce a dismisura. Per mantenere il valore esatto, una possibilità è quella di immagazzinarne le cifre in un vettore C(..): i prodotti parziali K*F possono essere calcolati moltiplicando ogni cifra di F per K ricostruendo poi le cifre del prodotto come nel seguente esempio:

Calcolare 842*13

C(…)	0	0	8	4	2	
					*13	
C(…)	0	0	104	52	26	: 26=2*10+6
				+2		
C(…)	0	0	104	54	6	: 54=5*10+4
			+5			
C(…)	0	0	109	4	6	:109=10*10+9
		+10				
C(…)	0	10	9	4	6	
	+1					
C(…)	1	0	9	4	6	

L'aggiornamento del vettore C(..) avviene in due passi. Il

valore contenuto in C(K) viene separato in due parti: U che contiene la cifra delle unità di C(K) e D che contiene la parte che precede U; se C(K)=725 avremo U=5 e D=72. Per il calcolo sarà: D=INT(C(K)/10) e U=C(K)-D*10 (infatti, continuando con l'esempio precedente, D=INT(725/10)=INT(72,5)=72 e U=725-72*10=5), dopodichè U diviene il nuovo C(K) e C(K+1)=C(K+1)+D. Naturalmente, il vettore C(..) sarà dimensionato col valore calcolato nel miniprogramma (per comodità non utilizzeremo C(0)).

Ecco allora il programma completo:

```
100 poke 53280,6:print chr$(5);chr$(147)
110 input"n = ";n:if n<4 then 110
120 for h=1 to n:s=s+log(h):next
130 nc=int(s/log(10))+1:print:print"il
numero di cifre di n! e' ";nc
140 print:print"procedo con il calcolo, abbi
pazienza":print
150 dim c(nc):c(1)=1
160 for h=1 to n:print"..";h;
170 for j=1 to nc:c(j)=c(j)*h:next j
180 for k=1 to nc-1
190 d=int(c(k)/10):u=c(k)-d*10
200 c(k)=u:c(k+1)=c(k+1)+d
210 next k:next h
220 print:print:print n;"! = ";
230 for w=nc to 1 step -1
240 print right$(str$(c(w)),1);
250 next w
```

L'istruzione 100 si incarica dei preliminari: colore bordo, colore caratteri, pulizia dello schermo e la 110 richiede il numero (maggiore di 3) di cui si intende calcolare il fattoriale.

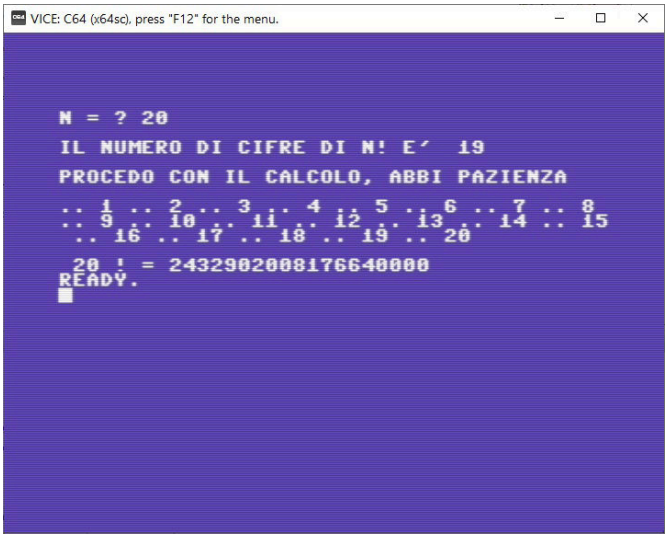
Nella 120 e 130 viene calcolato il numero delle cifre (NC) di "n!" e alla 150 il vettore C(..) viene dimensionato e inizializzato.

Alla 160 inizia il ciclo principale; il contatore H, che ci dice a che punto siamo con il calcolo, viene stampato in modo che l'utilizzatore possa farsi un'idea di quanto dovrà aspettare. All'interno del ciclo principale sono presenti altri due cicli "for-next": quello alla 170 moltiplica ciascuna cifra di C(..) per il K del momento, quello dalla 180 alla 210 esegue il "travaso delle cifre" nei termini esposti prima. Al crescere di "n", il numero delle iterazioni cresce a dismisura e i tempi d'attesa possono diventare proibitivi. Quando si arriva alla 220 il calcolo è stato completato: C(1) contiene la cifra delle unità, C(2) quella delle decine e così a salire fino a C(NC) che contiene la cifra più significativa.

Il ciclo 230-250 si incarica della stampa finale; il ricorso a RIGH$(STR$(C(W)),1) serve per eliminare lo spazio iniziale che il C64 pone automaticamente prima della cifra. In questo modo le varie cifre vengono stampate "incollate" l'una all'altra.

Un consiglio: all'inizio utilizzate il programma con piccoli valori di "n", n=10, n=20, poi ... vedete voi. L'interprete Basic del C64, lo sappiamo, è un po' un lumacone, il calcolo di 100! attraverso l'emulatore VICE ha richiesto ben 24 minuti.

Ecco un paio di output:



“Cielo... mio marito!”

di András Vajda

Sommario

La celeberrima battuta tratta da “*Tailleur pour dames*” del grande Georges Feydeau, una delle più esilaranti (e copiate) pochade incentrate sul tema delle relazioni extraconiugali, si presta splendidamente ad introdurre il non meno famoso problema combinatorio noto come “Problema dei matrimoni stabili”. In questo articolo proponiamo come primo assaggio una succinta descrizione del problema così come posto originariamente da Gale & Shapley nel 1962 e una elegante implementazione in COMAL 80 per Commodore 64.

Introduzione

Sfatiamo subito un mito: scorrendo l’ormai vastissima lista delle applicazioni nate attorno a questo problema e sue variazioni ([Man13], pagg. 30 ÷ 31), si vede come in realtà l’uso di tali algoritmi in ambito matrimoniale e simili (dating online, etc.) è sporadico, pressoché inesistente. Come spessissimo avviene, il riferimento al “matrimonio stabile” usato nel nome del problema è volutamente metaforico e serve unicamente a creare uno spiritoso spunto mnemonico e situazionale per quello che è un serissimo problema di assegnazione di risorse in operations research, nato attorno ad una istanza importante (l’assegnazione di studenti ai college universitari e, indipendentemente, dei medici tirocinanti alle cliniche) ed evolutosi poi in una serie di generalizzazioni con un vastissimo campo applicativo: dall’assegnazione di posti istituzionali in enti pubblici e parastatali di ogni genere alla logistica, dalla pianificazione della produzione all’assegnazione di risorse ai centri di costo, alla gestione di affiancamenti nell’ambito sanitario o militare, eccetera.

Matrimoni stabili: l’origine storica del problema

In cosa consiste esattamente il problema Stable Marriage (SM)? Esistono numerose varianti, ma nella formulazione originale, dovuta a Gale e Shapley nel 1962 ([GS62]), si hanno due distinti gruppi (uomini e donne), di pari dimensione. Ciascun membro di ogni gruppo esprime in una lista completa le proprie preferenze per l’altro sesso, ordinandole - per fissare le idee - in modo decrescente, dalla più alla meno desiderabile. In pratica, ad ogni elemento di uno dei due insiemi si associa una particolare permutazione dell’altro insieme. Si procede quindi a formare le coppie, seguendo gli elenchi di preferenze: la soluzione ottenuta è detta matrimonio stabile se non esistono due individui, uno per insieme, i quali preferiscono reciprocamente l’altro individuo al proprio partner attuale. L’algoritmo pubblicato nel 1962 (che chiameremo GS) è basato su semplicissimi passaggi detti "proposte", che vengono accettate o rifiutate secondo il semplice confronto posizionale tra il partner attuale e il proponente nella scala delle priorità della parte che riceve la proposta. Se la ricevente preferisce il proprio partner attuale al proponente, scatta il manzoniano "Questo matrimonio non s’ha da fare" e si passa alla successiva preferenza. L’idea tipica, nelle intenzioni di Gale & Shapley, è che l’insieme M degli "uomini" (es. studenti) si propone e quello delle "donne" W (es. college, cliniche) valuta le proposte, ma ovviamente nulla impedisce di eseguire l’algoritmo seguendo la convenzione opposta. Il principale merito di Gale & Shapley è quello di avere dimostrato che:

1. Qualunque istanza del problema SM ammette almeno una soluzione stabile;
2. Tale soluzione ha la proprietà di essere proponent-optimal, ossia garantisce a ciascun membro dell’insieme proponente il migliore accoppiamento in assoluto rispetto a qualsiasi altra eventuale soluzione stabile (e non). Questa proprietà è la cosiddetta ottimalità paretiana debole.

Il tutto usando un algoritmo di estrema semplicità che, dati due insiemi non vuoti M e W di cardinalità n e le relative matrici di preferenze, genera la soluzione male-optimal (o female-optimal) in O(n2) al caso peggiore. L’analisi del caso medio di esecuzione avrebbe diritto ad un intero articolo dedicato, ma ci limitiamo qui a presentarne il risultato che coinvolge la definizione di numero armonico:

$$H_n = \sum_{k=1}^{n} k^{-1} = \gamma + \psi_0(n+1)$$

dove Hn è appunto l'ennesimo numero armonico. Nell'espressione analitica più a destra, γ è la costante di Euler-Mascheroni e ψ0() è la funzione digamma, ossia la derivata logaritmica della funzione gamma Γ(), a sua volta estensione del fattoriale ai numeri reali. Si veda il classico [Leb72].

Detto questo, il numero medio di proposte nell'algoritmo GS è dato da n · Hn + O((logn)4). Ne consegue che la complessità media è dell'ordine di Θ(nlogn).

La soluzione ottenuta può essere ovviamente espressa in vari modi. Sostanzialmente essa consta di un elenco di n coppie univoche (m,w) con m ∈ M e w ∈ W. Si abbiano i seguenti due insiemi: M = {Aldo, Bruno, Carlo, Dario} e W = {Elena, Flora, Grazia, Helga}. Ciascun membro degli insiemi esprime il suo elenco di preferenze, usando le iniziali dei nomi:

Gentlemen				
A	H	E	F	G
B	F	G	E	H
C	F	H	G	E
D	G	E	H	F

Ladies				
E	D	A	C	B
F	A	C	B	D
G	A	B	C	D
H	D	A	C	B

Intuitivamente, la prima riga della matrice a destra ci dice che la signorina Elena preferisce nell'ordine il signor Dario, poi come seconda scelta il signor Aldo, e via di seguito in ordine discendente di preferenza.

Dal punto di vista computazionale, tuttavia, come nella maggior parte della letteratura, è invalsa l'abitudine di codificare gli insiemi di input usando piccoli interi sia per gli uomini che per le donne. Anche se inizialmente la notazione può creare una certa confusione, si tratta comunque di una codifica estremamente efficace, universale e facile da manipolare, soprattutto per grandi insiemi.

Così, le nostre tabelle assumono un aspetto leggermente diverso, mantenendo esattamente il medesimo significato:

Gentlemen				
1	4	1	2	3
2	2	3	1	4
3	2	4	3	1
4	3	1	4	2

Ladies				
1	4	1	3	2
2	1	3	2	4
3	1	2	3	4
4	4	1	3	2

La soluzione stabile male-optimal generata da GS è il seguente elenco di coppie che segue la convenzione (m,w): {(1,4),(2,3),(3,2),(4,1)}, che corrisponde a {(A,H),(B,G),(C,F),(D,E)} usando le rispettive iniziali. Tale elenco può ovviamente essere espresso come una permutazione in notazione standard a due linee:

Leggendo la notazione da sinistra a destra e dall'alto in basso, abbiamo appunto le coppie (1,4),(2,3),(3,2),(4,1). In questa notazione, la permutazione identità della prima riga rappresenta l'insieme proponente nell'ordine naturale. Ne consegue che si può usare anche la notazione posizionale 1-line, come per qualsiasi altra permutazione, senza perdita di informazione: la notazione 4 3 2 1 rappresenta esattamente le medesime coppie, dove agli uomini è implicitamente assegnata la posizione 1..n crescente verso destra e le relative partner vengono specificate esplicitamente.

Tuttavia, la natura combinatoria elementare della soluzione non ci è di grande aiuto computazionalmente. Trovare tra le possibili permutazioni dell'insieme W (o M, se del caso) la soluzione ottimale per esaustione applicando il paradigma del "generate and test" è una via del tutto impraticabile anche per problemi di dimensioni assai modeste. Tale soluzione può anche essere vista nella sua natura di relazione binaria, ossia un sottoinsieme del prodotto cartesiano M ×W tra i due insiemi dati, il quale prodotto in soldoni non è altro che l'elenco di tutte le possibili n2 coppie ordinate formate prendendo un solo elemento per volta da ciascun insieme di partenza. Assolutamente banale dal punto di vista combinatorio, in ultima analisi: ma come spesso avviene, gli algoritmi necessari ad ottenere una enumerazione esaustiva di tali soluzioni, come pure quelli dedicati a trovare particolari tipologie di soluzione, erano ben lungi dal vedere la luce ai tempi di Gale & Shapley, sono quasi tutti caratterizzati da complessità computazionale molto elevata e/o richiedono un approccio del tutto peculiare, con l'ausilio di strutture dati non elementari.

Gli anni Settanta e Ottanta

Pochi anni dopo la pubblicazione dell'articolo di Gale & Shapley, McVitie e Wilson nel 1970/71 [MW70, MW71b, MW71a] presentano vari algoritmi in ALGOL60, il più importante dei quali consente la generazione esaustiva di tutte le soluzioni stabili per una data istanza di SM (per la versione del problema generalizzata al caso di liste di dimensioni disuguali, che come caso particolare risolve l'istanza standard). Sfortunatamente, per motivi di copyright ancora vigente su tali articoli, non sarà possibile

qui approfondire come meriterebbe l'argomento presentando il sorgente originale integrale. Tale algoritmo ricorsivo, pur avendo prestazioni decisamente non entusiasmanti, ha una sua notevole importanza storica in quanto costituisce il primo tentativo di fornire una generazione esaustiva delle soluzioni. Negli stessi anni anche D. E. Knuth aveva proposto un algoritmo sostanzialmente simile, con analoghe prestazioni.

Occorreranno quasi venti anni prima che tale sforzo sia superato, con la pubblicazione di alcuni articoli poi confluiti nella fondamentale monografia di Gusfield e Irving [GI89] i quali - grazie ad un attento studio della natura algebrica dello spazio delle soluzioni come reticolo e come anello d'insiemi, e suggerendo un uso accorto di strutture dati order-preserving in grado di garantire contemporaneamente sia l'accesso diretto indicizzato, sia cancellazioni in O(1) - hanno proposto l'algoritmo ancora attualmente in uso per la generazione esaustiva anche su istanze di grandi dimensioni, con prestazioni in O(n2), praticamente il medesimo ordine di grandezza del GS (che in quel tempo è in grado però di generare solo una singola soluzione, non tutte). Avremo modo di approfondire altrove come si possano ottenere simili incrementi prestazionali.

Altro aspetto caratteristico (e divertente!) dell'algoritmo di McVitie e Wilson, che lo rende ancora più degno di studio e di interesse, è la sua originale implementazione su un KDF-9 a transistor della English Electric Computers (poi English Electric LEO Marconi Computers), installato presso l'Università di Newcastle: questo è genuino retrocomputing, e del migliore! Di tale storico elaboratore, dotato di memorie a nuclei magnetici, esiste un emulatore scritto in Ada e dotato di una ricchissima documentazione, col quale l'Autore si è a lungo divertito.

Come già accennato, a partire dalla metà degli anni Ottanta alcune pubblicazioni (principalmente dovute a Gusfield, Irving e Leather: [IL86, ILG87, GILS87, Gus87]) avevano risvegliato l'interesse generale verso il problema, stimolando numerosi articoli divulgativi sulle principali riviste internazionali di programmazione applicativa e - come immediata conseguenza - anche diverse implementazioni di studio in vari linguaggi di alto livello (tra i quali Ada, C, Clipper, COMAL) da parte dell'Autore. In questo articolo ci concentreremo su una di tali implementazioni in COMAL 80 V. 2.01 per Commodore 64.

Un'ultima nota. La prima volta che l'Autore si è occupato sistematicamente del problema in un contesto accademico, a fine anni Ottanta, esistevano decine di pubblicazioni al riguardo, SM era menzionato in almeno mezza dozzina di testi di algoritmica a partire dagli anni Settanta e oltre alla già citata Gusfield-Irving (allora fresca di stampa) vi erano altre due monografie dedicate, una delle quali [RS90] fondamentalmente incentrata solo sugli aspetti game-theoretic e l'altra, dovuta al venerabile D.E. Knuth [Knu97], edita solo in francese fino al 1997. Sei lustri dopo il numero di pubblicazioni e di testi di algoritmica che fanno menzione del problema è pressoché raddoppiato, tuttavia si deve rilevare che molte delle questioni poste nelle principali monografie storiche (alle quali se ne è aggiunta una quarta [Man13] pochi anni fa) non hanno ancora trovato soluzione e il problema rimane ricco di spunti di ricerca e applicativi.

Il lettore interessato può scaricare un articolo di approfondimento sull'algoritmo di Gusfield & Irving, latamente correlato al retrocomputing (viene proposta una implementazione in C89 conforme agli strumenti disponibili all'epoca della pubblicazione della monografia, senza fare ricorso a Python e strutture dati avanzate), la cui dimensione renderebbe difficoltosa la pubblicazione in questa sede, anche in numerose puntate.

L'algoritmo di Gale & Shapley

La versione più elementare, riportata nei testi di algoritmica e in numerosi articoli divulgativi, è anche la più utilizzata per le implementazioni. Si tratta di un algoritmo iterativo di estrema semplicità, che però garantisce un risultato per nulla intuitivo leggendo il problema. Il nondeterminismo introdotto dalla mancanza di un particolare ordine dei proponenti è un aspetto che è stato dato per scontato negli articoli scientifici degli anni Sessanta e Settanta: in realtà l'ordine dei proponenti è del tutto irrilevante, perché l'algoritmo genera sempre e comunque la medesima soluzione male-optimal (o female-optimal, invertendo banalmente le tabelle di preferenze), come non mancano di notare Gusfield e Irving ([GI89], pagg. 9 ÷ 10).

```
procedure STABLE_MARRIAGE
  assign each person to be free;
  while some man m is free do
  begin
     w:=next(m);
```

```
    if free(w) then
        assign m and w to be engaged {to each
other};
    else
          if w prefers m to her fiancé m' then
            assign m and w to be engaged;
            free m';
          else
             w rejects m {m remains free};
  end;
  output the n engaged pairs;
```

Il funzionamento dell'algoritmo è decisamente intuitivo. La funzione di comodo next(m) fornisce la donna w immediatamente successiva all'attuale partner (o la prima della lista, se siamo alla prima esecuzione) nell'elenco di preferenze dell'uomo m, il proponente. La funzione free(w), in modo del tutto ovvio, controlla se w ha già un partner. In tale caso, si confrontano le posizioni del proponente m e dell'attuale partner m' nell'elenco delle preferenze di w e si agisce di conseguenza, realizzando la coppia che assegna a w il partito relativamente migliore secondo le preferenze di lei e libera il meno desiderabile, che così potrà proporsi ad altre potenziali partner nelle successive iterazioni. Importante rilevare che tali comparazioni, così come l'implementazione delle funzioni ausiliarie, possono e devono essere realizzate in O(1) senza ricorso a farraginose ricerche lineari negli array principali, data anche la dimensione media dei problemi generalmente trattati con questa famiglia di algoritmi, dell'ordine di migliaia di proponenti. Altrettanto importante notare che i controlli di bound sugli indici sono sostanzialmente superflui, in quanto Gale & Shapley hanno dimostrato che l'algoritmo termina sempre senza che alcun proponente oltrepassi il termine della propria lista di preferenze, restando così celibe.

L'idea vincente dell'algoritmo si può spiegare in modo semplicissimo: ogni conflitto di preferenze tra i proponenti viene risolto in modo deterministico, scegliendo tra i due quello che risulta più vicino alla cima della lista preferenziale della donna contesa. In questo modo uno o più proponenti dovranno accontentarsi di una partner che preferiscono meno, come mostra l'esito finale dell'esempio che vedremo tra breve, ma che comunque (è dimostrabile) rimane la loro migliore scelta in assoluto tra tutte le possibili soluzioni stabili.

Chiudiamo questa breve sezione con una curiosità storica: nelle implementazioni fino a tutti gli anni Settanta, veniva usato il bit del segno per indicare se un individuo m o w risultava coniugato, senza utilizzare un array ausiliario. Un tipico trucco di retroprogrammazione che ha caratterizzato, prima ancora dell'era dei PET e degli home computer, un'intera epoca di programmi in ALGOL e FORTRAN, condizionata dai costi delle memorie fisiche e dalla loro proverbiale scarsità.

L'implementazione in COMAL 80

Si presenta una implementazione di esempio in COMAL 80 (provata con la versione 2.01 su cartridge) per Commodore 64, versione semplificata e ridotta di una delle prime implementazioni dell'Autore, datata 1984. Per verificare facilmente la correttezza dell'implementazione, si sono scelte le tabelle di preferenze dell'esempio di dimensione 8 da [GI89], pag. 12 che generano la soluzione stabile male-optimal seguente, ordinando al solito le coppie secondo la convenzione (m,w):
{(1,5),(2,3),(3,8),(4,6),(5,7),(6,1),(7,2),(8,4)}

```
//*********************************
// save stable-marriage
//*********************************
DATA 5,7,1,2,6,8,4,3
DATA 2,3,7,5,4,1,8,6
DATA 8,5,1,4,6,2,3,7
DATA 3,2,7,4,1,6,8,5
DATA 7,2,5,1,3,6,8,4
DATA 1,6,7,5,8,4,2,3
DATA 2,5,7,6,3,4,8,1
DATA 3,8,4,5,7,2,6,1

DATA 5,3,7,6,1,2,8,4
DATA 8,6,3,5,7,2,1,4
DATA 1,5,6,2,4,8,7,3
DATA 8,7,3,2,4,1,5,6
DATA 6,4,7,3,8,1,2,5
DATA 2,8,5,3,4,6,7,1
DATA 7,5,2,1,8,6,4,3
DATA 7,4,1,5,2,3,6,8

DIM male'pref(8,8)
DIM female'pref(8,8)
DIM male'lut(8,8)
DIM m'coupled(8)
DIM w'coupled(8)
DIM stack(8)
```

```
stack'ptr:=0

PAGE
FOR i:=1 TO 8 DO
  FOR j:=1 TO 8 DO
    READ male'pref(i,j)
  ENDFOR j
  push(i)
ENDFOR i

FOR i:=1 TO 8 DO
  FOR j:=1 TO 8 DO
    READ female'pref(i,j)
    male'lut(female'pref(i,j),i):=j
  ENDFOR j
ENDFOR i

disp'data

WHILE stack'ptr>0 DO
  pop(man)
  m'coupled(man):+1
  woman:=male'pref(man,m'coupled(man))
  IF w'coupled(woman)<1 THEN
    w'coupled(woman):=man
  ELSE
                                          IF
male'lut(man,woman)<male'lut(w'coupled(woman)
,woman) THEN
      push(w'coupled(woman))
      w'coupled(woman):=man
    ELSE
      push(man)
    ENDIF
  ENDIF
ENDWHILE

disp'solution

END "Fine lavoro"

PROC disp'data
  PRINT "Uomini:              Donne:"
  FOR i:=1 TO 8 DO
    PRINT i,": ",
    FOR j:=1 TO 8 DO
      PRINT male'pref(i,j)," ",
    ENDFOR j
    PRINT AT i+1,21: i,": ",
    FOR j:=1 TO 8 DO
      PRINT female'pref(i,j)," ",
    ENDFOR j
    PRINT
  ENDFOR i
ENDPROC disp'data

PROC disp'solution
  PRINT AT 12,1: "Stable marriage (m,w):"
  FOR i:=1 TO 8 DO
    w:=male'pref(i,m'coupled(i))
    PRINT AT 12+i,1: "(",i,",",w,")"
    PRINT AT i+1,2+2*m'coupled(i): CHR$(158),w
          PRINT  AT  w+1,22+2*male'lut(i,w):
CHR$(159),i,CHR$(154)
  ENDFOR i
  PRINT AT 20,1: CHR$(154)
ENDPROC disp'solution

PROC push(d)
  stack'ptr:+1
  stack(stack'ptr):=d
ENDPROC push

PROC pop(REF d)
  d:=0
  IF stack'ptr>0 THEN
    d:=stack(stack'ptr)
    stack'ptr:-1
  ENDIF
ENDPROC pop
```

Il listato, sebbene privo di commenti, risulta del tutto autoesplicativo. Si è scelto di implementare uno stack nel modo più classico, tramite l'array denominato (con audace spicco di fantasia) stack() e la singola variabile stack'ptr. Il valore 0 è in questo caso perfetto come guard value, poiché in COMAL gli indici partono sempre da 1. Si noti che in COMAL il carattere apostrofo è perfettamente valido se utilizzato come separatore nei nomi di procedura e variabile, esattamente come l'underscore in altri linguaggi. Gli array male'pref() e female'pref() contengono, senza alcuna sorpresa, le liste delle preferenze dei due sessi, mentre male'lut() è utilizzato per il reverse lookup in tempo costante, poiché per ogni possibile coppia (m,w) ci fornisce la posizione di m nella lista delle preferenze di w, rendendo possibile il confronto immediato tra la posizione del proponente e quella dell'attuale partner, come richiesto dall'algoritmo.

Ultima banale nota: l'array w'coupled() ci dice in modo immediato chi è attualmente il partner di w per ogni w

W e le sue locazioni contengono ciascuna un piccolo intero $1 \leq m \leq 8$ che identifica il partner, mentre l'array m'coupled(), come è evidente dal codice, è utilizzato in

modo leggermente diverso. Il contenuto di ciascuna cella è sempre un piccolo intero non negativo, ma usato stavolta come indice relativo alla posizione di w nella lista delle preferenze di m. In questo modo, ad esempio, considerando la coppia (m,w) = (1,5) si avrà che m'coupled(m)=1 in quanto male'pref(m,1)=5, e così via per ogni altra coppia. A proposito di array, si ricordi solo che essi sono implicitamente inizializzati a zero, così come le variabili in genere. Si noti ancora una volta come il linguaggio è un po' prolisso (come ALGOL, COBOL o FORTRAN) ma estremamente chiaro, leggibile, ordinato. Il risultato finale è visibile nello screenshot che segue, dove i partner sono elencati esplicitamente ed evidenziati con colori diversi nelle matrici di preferenze dalla procedura disp'solution, mostrando la collocazione relativa dei partner in ciascuna lista di preferenze per stimolare doverose riflessioni nel lettore sul significato profondo dell'ottimalità paretiana dell'algoritmo.

Conclusioni

Si è presentato in estrema sintesi uno dei problemi combinatori più longevi e famosi, con qualche fondamentale riferimento alle principali tappe nello sviluppo delle soluzioni fino a tutti gli anni Ottanta. Per un più completo orientamento nella vasta letteratura esistente si rimanda senz'altro il lettore interessato alle monografie già citate, alla loro bibliografia ed al più recente survey disponibile: [IM08].

Si è quindi fornita una completa implementazione in COMAL 80 2.01 per Commodore 64 che consente di esemplificare le caratteristiche e la grande eleganza di tale linguaggio evoluto, rendendone trasparente il funzionamento grazie alla semplicità dell'algoritmo. La duplice speranza dell'Autore è che da un lato i lettori siano incuriositi dal problema e dalle sue numerose varianti, dall'altro che apprezzino sempre meglio il linguaggio e la sua grande espressività.

Riferimenti bibliografici

[GI89] Dan Gusfield and Robert W. Irving, "The stable marriage problem: Structure and algorithms", MIT Press, Cambridge, MA, USA, 1989.

[GILS87] Dan Gusfield, Robert Irving, Paul Leather, Michael Saks, "Every finite distributive lattice is a set of stable matchings for a small stable marriage instance", Journal of Combinatorial Theory, Series A 44 (1987), no. 2, 304-309.

[GS62] D. Gale, L. S. Shapley, "College admissions and the stability of marriage", The American Mathematical Monthly 69 (1962), no. 1, 9-15.

[Gus87] Dan Gusfield, "Three fast algorithms for four problems in stable marriage", SIAM Journal on Computing 16 (1987), no. 1, 111-128.

[IL86] Robert W. Irving, Paul Leather, "The complexity of counting stable marriages", SIAM Journal on Computing 15 (1986), no. 3, 655-667.

[ILG87] Robert Irving, Paul Leather, Dan Gusfield, "An efficient algorithm for the "optimal" stable marriage", J.ACM 34 (1987), 532-543.

[IM08] Kazuo Iwama, Shuichi Miyazaki, "A survey of the stable marriage problem and its variants", Proceedings of the International Conference on Informatics Education and Research for Knowledge-Circulating Society (Icks 2008) (USA), ICKS08, IEEE Computer Society, 2008, pp. 131-136.

[Knu97] Donald E. Knuth, "Stable marriage and its relation to other combinatorial problems", American Mathematical Society, Providence, R.I, 1997.

[Leb72] N. N. Lebedev, "Special functions and their applications," Dover Publications, Inc., New York, 1972.

[Man13] David Manlove, "Algorithmics of matching under preferences", World Scientific Publishing Company, 2013.

[MW70] D. G. McVitie, L. B. Wilson, "Stable marriage assignment for unequal sets", BIT Numerical
Mathematics 10 (1970), no. 3, 295-309.

[MW71a] D. G. McVitie, L. B. Wilson, "The stable marriage problem", Communications of the ACM 14 (1971), no. 7, 486-490.

[MW71b] D. G. McVitie, L. B. Wilson, "Three procedures for the stable marriage problem", Communications of the ACM 14 (1971), no. 7, 491-492.

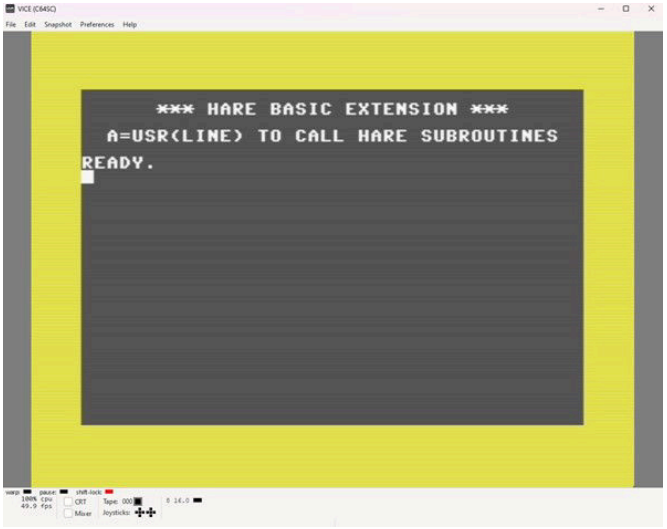
[RS90] Alvin E. Roth, Marilda A. Oliveira Sotomayor, "Two-sided matching: A study in game-theoretic modeling and analysis", Econometric Society Monographs, Cambridge University Press, 1990.

Hare Basic – 10 Volte più veloce

di Carlo Nithaiah Del Mar Pirazzini

Hare Basic nasce dalla mente di Aleksi Eeben, già autore del Bunny Basic nel 2019 con due scopi: festeggiare i 60 anni della nascita del BASIC e realizzare un interprete di base veloce e funzionale da poter sostituire in Basic 2.0 di C64 e VIC20.
Hare Basic è un sottoinsieme altamente ottimizzato del Basic Commodore che si può attivare o disattivare a seconda delle necessità. È un interprete e non un compilatore ed è dotato di pochi comandi e funzioni, tutti eseguibili direttamente su C64.

Ecco come si presenta HARE BASIC all'avvio:



La sintassi è molto simile, ma ci sono alcune restrizioni e differenze.
Non sono presenti variabili stringa, tuttavia esiste un supporto semplice per le stringhe con terminazione NULL che possono farlo.
Non ci sono ARRAY. Si accede alla memoria direttamente con POKE e PEEK per creare strutture dati più grandi.
È possibile una solo operazione aritmetica alla volta.
Non è possibile utilizzare gli spazi bianchi. Riceverai un messaggio ? HARE ERROR READY.
GOTO e GOSUB possono prendere una variabile come destinazione.
THEN non accetta solo una variabile numeri. Tuttavia, THENGOTO funziona con variabili.
La funzione STR$ viene riproposta per fare riferimento a stringhe con terminazione NULL nella memoria. Si possono utilizzare variabili come puntatore o utilizzare un indirizzo di memoria diretto.
RND è implementato come un comando che inserisce un numero casuale nell'intervallo 0-65534.
Il comando LOAD non riavvia il programma, ma continua con l'istruzione successiva.

Un listato d'esempio presente nella cartella di documentazione del HARE BASIC:

Ci sono anche molte altre differenze tutte spiegate con dovizia e numerosi esempi sul manuale presente nel sito https://csdb.dk/release/?id=242880 che vi consigliamo caldamente di leggere.

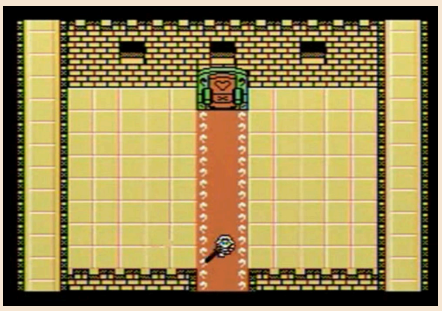
Nella pagina dropbox dell'autore (https://www.dropbox.com/scl/fo/qkn7ehgoctgcq4ldrlx43/AHwpvxdq06t3rVN9dkCrY4g?rlkey=t44bux2c3qtsa31t5eupank8t&e=2&st=0oe22jyb&dl=0) è presente anche una serie di esempi utili per comprendere la velocità di esecuzione e le differenze.

I numerosi esempi presenti nella dotazione del software

In conclusione mi sono divertito a "giocare" con questa versione snella e veloce del Basic, una scoperta e sicuramente una ventata di novità e freschezza, segno di una scena di retro sviluppo viva e vegeta.
Nei prossimi numeri analizzeremo assieme ad alcuni collaboratori alcuni listati sviluppati con HARE BASIC, non mancate!

QUEST FOR THE GOLDEN CHALICE

Anno: 2012
Editore/Sviluppatore: Team Pixelboy
Genere: Avventura
Piattaforma: Colecovision
Sito web: https://www.teampixelboy.com/quest_chalice.php



Draghi, Castelli, pipistrelli ed un principe codardo… mio Dio!
Tutto questo è Quest for the Golden Chalice, progetto del 2012 per Colecovision.
Un gioco che si ispira ad Adventure per Atari 2600; titolo molto amato dagli sviluppatori e si vede nel layout del gioco.
Ma questo Quest for the Golden Chalice è uno di quei titoli di cui sentivamo il bisogno sulla console di Coleco.
Grazie alle capacità grafiche della console rispetto all'Atari 2600, questo "omaggio" ad Adventure prende uno stile grafico ed un gusto moderno che ricorda la saga di "Zelda".
Un gioco divertente con una trama bislacca che vede il principe di Larosia, noto per la sua codardia, affrontare mostri, creature malvagie e dei potenti draghi.
Il calice dorato è stato rubato da un malvagio mago chiamato Mardok e nascosto.
Solo la spada, una buona dose di fortuna e diverse chiavi sparse qua e là permetteranno al principe di Larosia di ritrovare il calice e riportarlo al castello Giallo.
La struttura di gioco è semplice come era quella di Adventure. Girovagare per le mappe labirintiche, recuperare le chiavi ed evitare di essere uccisi da mostri terrificanti.
L'aspetto visivo è ben fatto e ricorda molto Zelda. La grafica è nitida e tutto si muove bene.
Insomma siamo di fronte ad un titolo che deve essere recuperato assolutamente e che porta sulle spalle solo 12 anni dalla sua uscita.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

» Giocabilità 85%
Un grande omaggio ad Adventure per Atari 2600 ma con lo stile alla Zelda. Semplice e divertente.

» Longevità 80%
Anche qui sono presenti diversi Easter Eggs come nel titolo originale.
Divertente.

NEW GAME

TIMO'S CASTLE

Anno: 2022
Editore/Sviluppatore: Roman Warner
Genere: Platform
Piattaforma: Commodore 64
Sito web: https://romwer.itch.io/hc

Timo's Castle è un prodotto ben fatto. Dietro al gioco si trovano un team rodato e una ottima programmazione. Nel gioco impersoneremo Timo McClane, giovane rampollo nobile scozzese. Un ragazzo che non vuole saperne della nobiltà e delle tradizioni, ma che si trova suo malgrado ad indagare sulla morte dei suoi genitori avvenuta in modo misterioso.

Da questo tragico evento abbiamo ereditato un piccolo maniero perduto nelle Highlands scozzesi.

Assieme alla fidanzata Amber, Timo si trova al suo interno con lo scopo di rimetterlo in ordine e, magari, di scoprire il destino della sua famiglia. Ma qualcosa da subito non va. Il castello è maledetto e una perfida strega rapisce Amber.

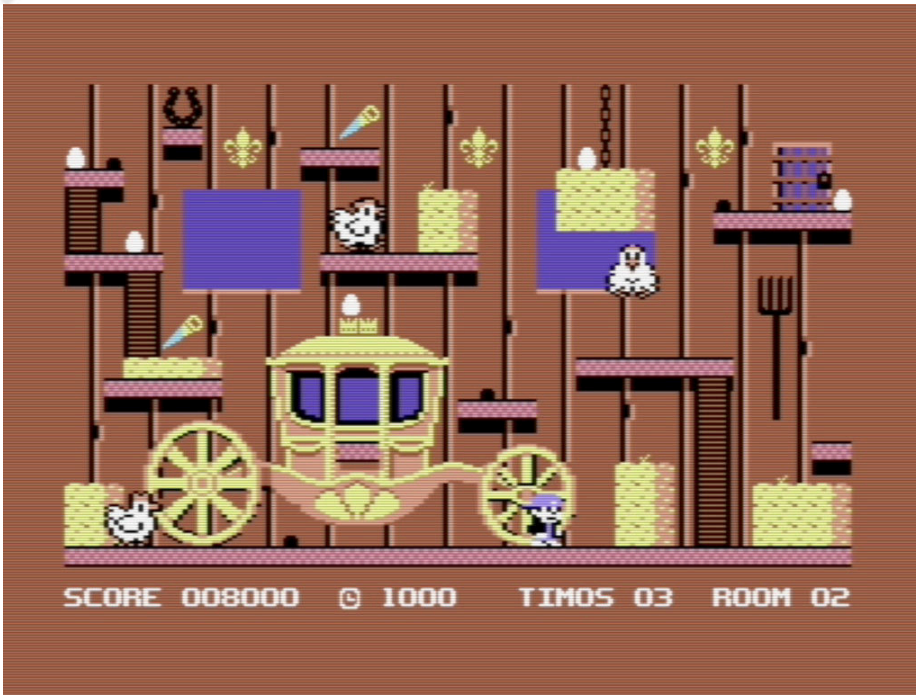
E la nostra storia ha inizio.

Timo's Castle è uno omaggio al famoso Henry's House. Come l'illustre predecessore è un platform a schermo fisso.

I livelli di gioco sono le stanze del castello (in totale 8) e sono tutte caratterizzate in modo differente e piene di oggetti posseduti che hanno la voglia di fare a pezzi il protagonista. Tutto questo immerso in tantissimi riferimenti familiari e ad un'atmosfera che ricorda i mitici anni 70, i funambolici 80 e i "grunge" anni 90. Timo's Castle mantiene tutti gli elementi che rendevano Henry House speciale. Le stanze ben realizzate e con una discreta dose di difficoltà. I nemici sempre pronti ad ostacolare il nostro eroe e quella voglia di vedere cosa c'è dopo ogni livello.

Questo legame è ovviamente migliorato: abbiamo un aspetto più moderno, animazioni più veloci e una cura per i dettagli maggiore.

L'aspetto grafico è gradevolissimo ed è apprezzabile lo sprite del protagonista che ci ha ricordato l'eroe



GIUDIZIO FINALE

» Giocabilità 90%
Mantiene lo stile di Henry's House nello stile dei livelli ma si lascia maneggiare meglio. Le collisioni e il design dei livelli sono bellissimi.

» Longevità 85%
Risente dello stile "old" di un tempo e richiede pratica, ma portarlo al termine è una grande soddisfazione.

di Magic Pockets, con quello strambo aspetto da "tamarro" degli anni 80.
Ma è il sonoro che abbiamo amato di più. Tutte le stanze sono caratterizzate da una bella colonna sonora e l'intro presenta una versione SID di "Scottland the Brave" perfetta.
Lo stile di Henry's House si sente. Non è un titolo semplice, soprattutto se preso sottogamba. Richiede precisione e attenzione per concludere i livelli.
Arrivare alla schermata finale non è impresa facile, ma la voglia di vederla vi farà premere sul tasto fire fino alla fine.

Il gioco è disponibile al download al costo di 4,99 dollari e li vale tutti. Non sarà l'assoluto capolavoro ma è un titolo dannatamente divertente.

di **Carlo Nithaiah Del Mar Pirazzini**

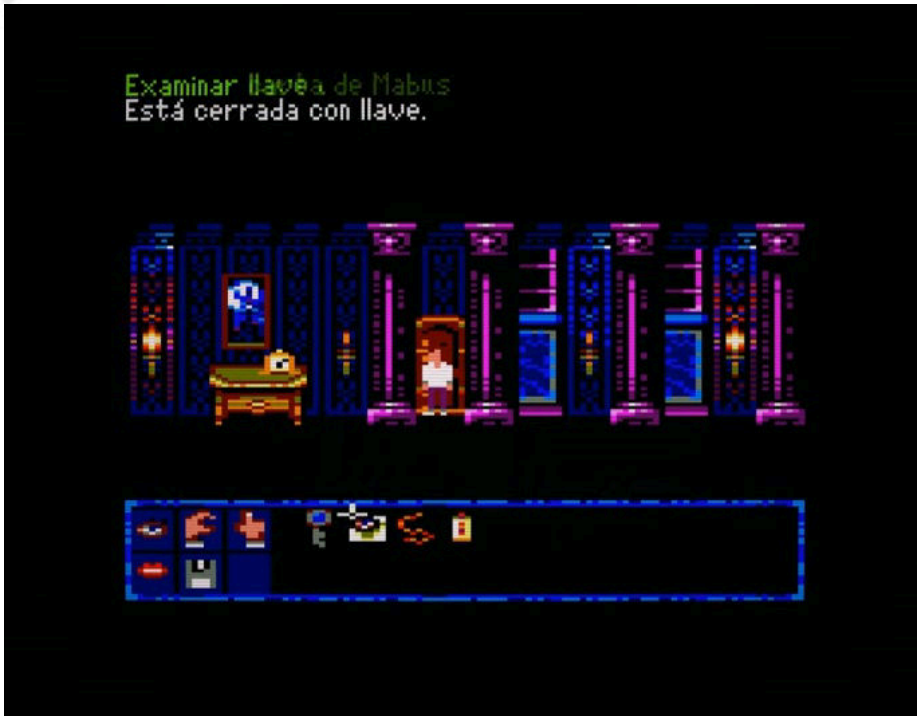
NEW GAME

THE KEY

Anno: 2024
Editore/Sviluppatore: Pakete Soft
Genere: Avventura punta e clicca
Piattaforma: Amstrad CPC
Sito web: https://www.paketesoft.com/es/thekey

Un fatto terribile compiuto quarant'anni prima, alcune cassette sono a testimonianza delle atrocità avvenute, un biglietto e una chiave affissi alla porta di casa e una nota inquietante sui misteri di Carfax Mansion.

Questi sono gli elementi di apertura del primo episodio di The Key, avventura punta e clicca in terza



GIUDIZIO FINALE

» Giocabilità 85%
Lo stile LucasArts è un plus per un'avventura punta e clicca dalla trama solida.

» Longevità 80%
Enigmi molto divertenti.

persona per Amstrad CPC.
Il team di sviluppo, i Pakete soft, nel 2019 rilasciarono già un titolo punta e clicca di buona fattura chiamato Escape the ROM, nato per mostrare il Paket engine, ma con The Key ci mettono davanti ad un vero e proprio omaggio allo spirito delle avventure "LucasArts" dei bei tempi andati.
La trama scorre bene ed è ricca di puzzle (alcuni davvero machiavellici!) e si lascia giocare senza portarci mai verso punti morti.
L'interfaccia ci ricorda alla lontana lo Scumm e funziona alla meraviglia, contornata da un aspetto tecnico ben fatto con una dettagliata e colorata grafica ed un buon accompagnamento musicale.
La bellezza di questo titolo parte già dal suo modo per essere scaricato... ovvero tramite un enigma!
Nel sito ufficiale dei Pakete si trova un folle enigma cliccabile che solo se risolto vi porterà a scaricare il titolo. Geniale!
La trama è interessante e ben sviluppata con alcuni tocchi di genio nei dialoghi. Il gioco è disponibile in lingua spagnola ma sono previste a breve due versioni: una in lingua francese ed una in inglese. Tirando le somme non mi resta che consigliarvi il titolo e suggerirvi di seguire il team Pakete Soft.

di **Giampaolo Moraschi**

NEW GAME

LYLE IN CUBE SECTOR

Anno: 2024
Editore/Sviluppatore: Michael Moffit
Genere: Platform/Metroidvania
Piattaforma: Sega Megadrive
Sito web: https://mikejmoffitt.itch.io/lics-md

"Il mio nome è Lyle. Purtroppo non posso dirvi il mio cognome; gli eroi dei videogiochi di solito hanno nomi composti di una sola parola ed io non voglio fare l'eccezione.

Inoltre, come la maggior parte degli eroi dei videogiochi, posso saltare più in alto del normale, la mia testa è più grande del mio busto (diamine anche i miei occhi sono più grandi!) e la mia salute non è determinata dalla pressione, dallo stress, dal cibo o da ciò che mi circonda, ma da una semplice barretta in alto sullo schermo che stai guardando.

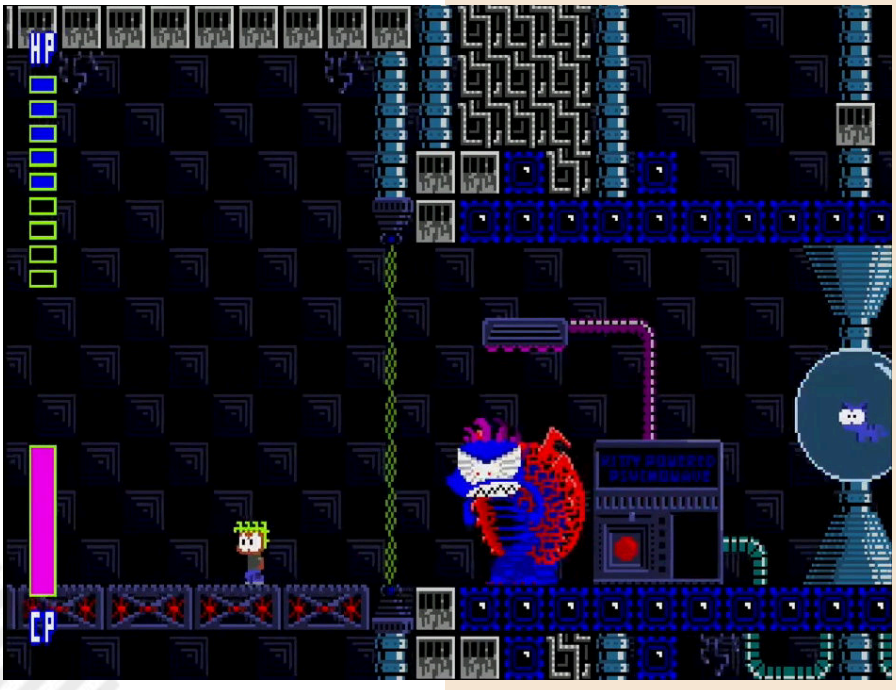
Vivo in una piccola baracca nel mezzo del settore Cubo, un regno popolato da ogni sorta di mostri cattivi. E sì! Ci sono tantissimi cubi qui in giro: verdi durevoli, rossi esplosivi, cubi arancioni di doppia dimensione e così via. In effetti, ho un cubo verde proprio nel mio cortile.

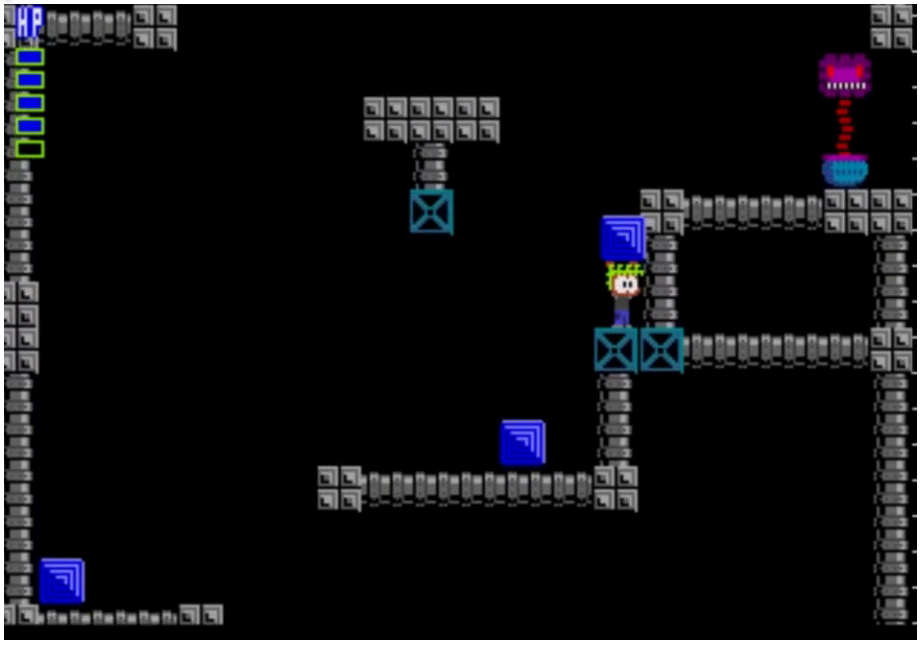
Il mio unico compagno di vita è un gattino di nome Keddums. Sul serio! Ora però basta parlare di me... se scarichi la rom puoi leggere tutto sul file txt che racconta del gioco. Uffa!

La scorsa notte, mentre dormivo, una figura incappucciata è entrata nel cortile e mi ha rapito il gatto. Inutile dire che questa cosa la trovo intollerabile: Keddums è il mio unico amico! Quindi, mentre scrivo queste righe, ho deciso di partire per reclamare il mio peloso e cubico amico.

Però non è facile questa avventura... cavolo appena esco di casa sono indifeso come un vermicello. Non posso nemmeno saltare in testa ai nemici come fa quel baffuto idraulico italo-nipponico. Devo trovare potenziamenti che mi migliorino. Ovviamente tutti incentrati sulla cubosità del gioco. Però è divertente dai. Posso lanciare cubi, sollevare cubi, calciare cubi e attaccare i nemici con i cubi.

Alla fine è una vita del cubo ma è divertente."



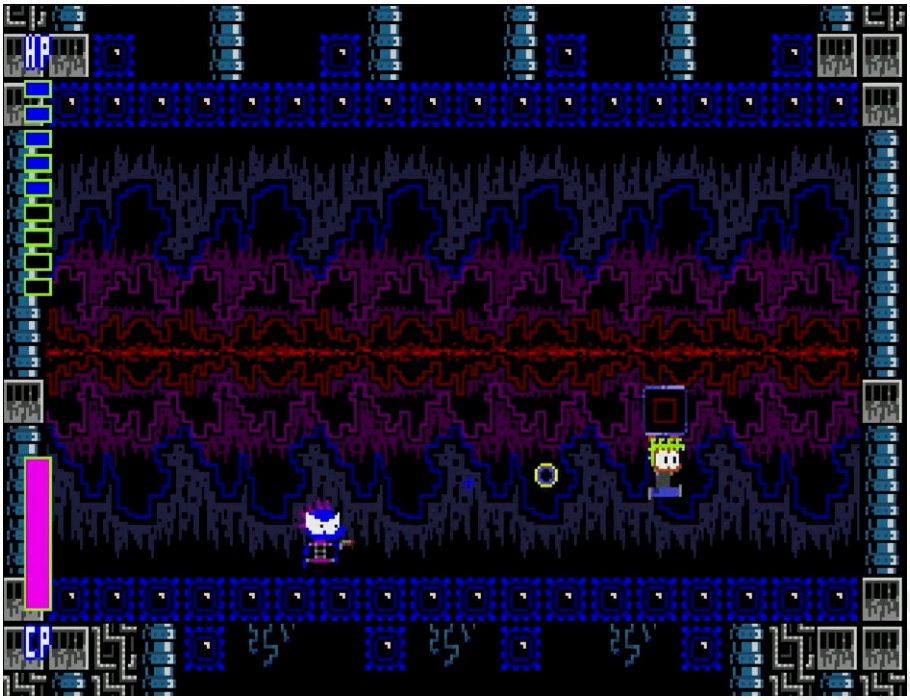


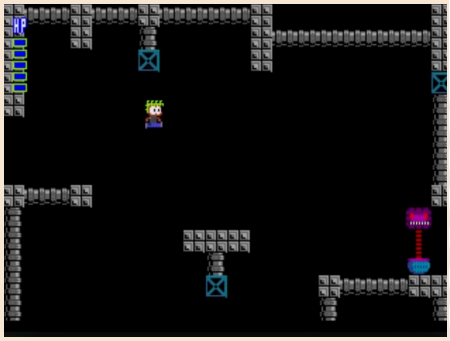
GIUDIZIO FINALE

» Giocabilità 90%
Vasto ma godibile in alcuni punti. Difetta per mancanza di semplicità iniziale.

» Longevità 75%
Alcune volte ci si trova a vagare senza meta.
Inizialmente non è facile rimanere in vita.

Lyle in Cube Sector è un gioco apparso qualche tempo fa su pc e solo ora vede una versione per Sega Megadrive praticamente identica ma decisamente più "godibile" per velocità.
È un gioco in stile Metroid. Il classico Metroidvania che va tanto di moda di questi tempi. Condivide il sistema esplorativo del famoso gioco Nintendo ma fortunatamente è meno "vasto" e permette un'esplorazione più mirata.
Possiede anche una forte dose di umorismo e un simpatico e curato aspetto grafico piuttosto minimalista che sulla console Sega ci sta decisamente bene.
Non è un titolo perfetto, ovviamente. In alcuni punti presenta una dispersione di trama e offre poche indicazioni su come muoversi nell'avventura. Questo ci porterà ad esplorare più e più volte gli stessi posti e, alla lunga, potrebbe annoiarci.
Altro appunto è la difficoltà iniziale davvero "tenace". Partire senza potenziamenti e debolissimi ci porterà spesso a game over. Avrei dosato meglio tutto questo con qualcosa di più fruibile.
Ma nel complesso, questo gioco merita di essere scaricato e apprezzato.
C'è tanto da esplorare e tiene occupati per un po' di tempo. La sequenza finale è divertente.
di **Carlo Nithaiah Del Mar Pirazzini**





ATLANTEAN

Anno: 2024

Editore/Sviluppatore: Aetherbyte

Genere: Shoot em Up

Piattaforma: Pc Engine

Al giorno d'oggi il PC Engine è tutt'altro che una nota a piè di pagina nella storia dei videogiochi per molte persone.

I veri amanti dei videogiochi non possono fare a meno di apprezzarlo come una delle piattaforme classiche più affascinanti e avere un nuovo sparatutto rilasciato 20 anni dopo la sua scomparsa è a dir poco eccezionale.

Sviluppato da un piccolo studio indipendente nel 2014, Atlantean è a tutti gli effetti il risultato di un lavoro d'amore mirato alla base di fan hardcore, come molti altri esempi che l'hanno preceduto negli ultimi anni.

Atlantean non prova nulla di nuovo o rivoluzionario, e a prima vista potrebbe essere scambiato per uno pseudo-sequel dell'universalmente impegnato Deep Blue . Fortunatamente l'unica cosa in comune tra i due giochi è il tema sottomarino, con il gameplay reale molto diverso tra loro. L'omaggio casalingo a Defender è chiaramente evidente e dal momento che non esiste un gioco con uno stile equivalente nella libreria di PC Engine, possiamo tutti salutarlo come il primo del suo genere per la console per videogiochi di NEC. Certo, ci sono voluti solo più di 20 anni per avere un nuovo titolo, ma ora che è qui immagino che alcune persone avranno un assaggio della vecchia corsa al gameplay di Defender .

Comunque, la missione del giocatore è proteggere gli abitanti sottomarini del pianeta Atlantide dall'essere rapiti dagli invasori robotici Aquanoid per scopi indicibili.

Per portare a termine il compito eroico si può sparare (pulsante II), attivare bombe che puliscono lo schermo (pulsante I) e muoversi

GIUDIZIO FINALE

» Giocabilità 80%
Tecnicamente non il massimo ma decisamente giocabile anche se soffre di alcuni rallentamenti.

» Longevità 80%
Un buon livello di difficoltà.

orizzontalmente ad ipervelocità (START). Il campo di gioco è quello già visto in Defender e nei suoi mille cloni, la velocità di scorrimento è dettata dal giocatore e un radar in alto ti dice la posizione di tutti i nemici e anche degli atlantidei che dovresti proteggere dal rapimento. Girare a sinistra ea destra deve tenere in considerazione la continua combinazione di velocità e inerzia.

Ogni fase è composta da cinque sezioni. Nella prima si affronteranno tre ondate di invasori consecutive, nella seconda saremo al centro dello schermo per attraversare le fessure delle pareti della miniera in arrivo e nella terza un'altra ondata di invasori. Poi si arriva al boss "aqualord", seguito da una sezione bonus in cui è necessario catturare atlantidei che cadono per avere l'opportunità di ripopolare gli abitanti rapiti o morti. Questo round bonus è assente solo nel 4° e ultimo livello, poiché il gioco termina quando il 4° Aqualord viene sconfitto.

Suoni e grafica sono abbastanza nella media per gli standard di PC Engine. La totale mancanza di potenziamenti è un po' deludente, ma almeno il gameplay di base è sufficiente per mantenere vivo l'interesse. Gli strati di parallasse sono usati ovunque e possono indurre un po' di scatti (in particolare nella sezione delle mine), mentre il design colorato sembra monotono e non consente a nessun palco di distinguersi dagli altri. C'è una Statua della Libertà distrutta da qualche parte nel gioco, ma cosa ci fa esattamente in un pianeta alieno? Scherzi a parte, il gioco non riesce ad andare oltre il livello base di realizzazione.

Nel complesso, l'unico vero svantaggio del gameplay vero e proprio è il notevole rallentamento quando incontri troppi nemici contemporaneamente, il che rende brevemente i controlli non reattivi e influisce sulle funzioni di fuoco/bombardamento.

Tirando le somme si tratta di un titolo forse non eccelso tecnicamente ma divertente.

Non si inserisce tra i migliori shooter per la console, ma merita una partitella ogni tanto.

di **Carlo Nithaiah Del Mar Pirazzini**

DICING KNIGHT PERIOD

Anno: 2004
Editore/Sviluppatore: Qute/ Platine Dispositif
Genere: ARPG
Piattaforma: WonderSwan Color

Dicing Knight Period è un gioco molto intelligente. Ora, non intendo questo nel senso di trama profonda, intricati meccanismi di gioco e sapienti indovinelli in game. In realtà, non è altro che un umile gioco d'azione rilasciato verso la fine del ciclo vitale della console portatile WonderSwan (abbiamo parlato di questa sfortunata console nei numeri 37 ita e 15 uk di RetroMagazine, andate a recuperarli, NdR).

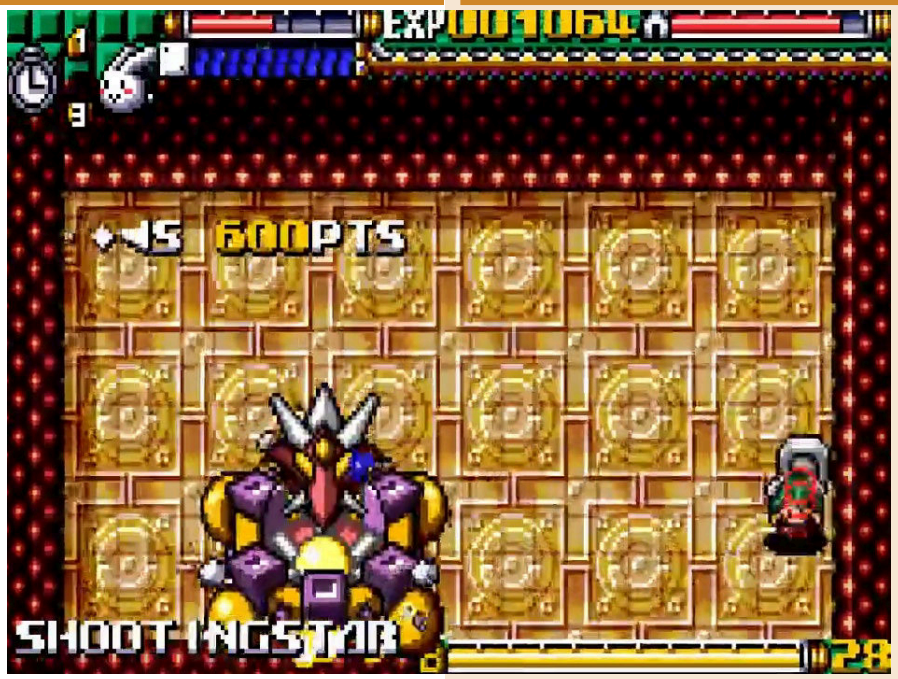
Ma è il modo in cui Dicing Knight racchiude quell'azione che lo rende così intelligente.

Il WonderSwan era un sistema portatile dal design molto particolare e questo gioco lo riflette.

Dalle piccole stanze all'azione è sorprendete quanto questo titolo dimostri l'unità tra la piattaforma e il suo design. È un'esperienza molto più forte di quanto inizialmente lasci intendere.

Ma cos'è esattamente Dicing Knight? Chiamarlo Action Game non dice molto a ciò che è e come funziona. Invece è meglio metterlo tra un mix di gameplay: quello d'azione di Final Fantasy Adventure e un'esplorazione profonda dei dungeon alla roguelike. Si oscilla la spada contro i vari mostri e si esplora un dungeon, stanza per stanza, piano per piano.

Ovviamente, non c'è molto da dire in un gioco del genere. Non richiede chissà che abilità per essere giocato e, con solo due pulsanti con cui interagire, risulta un gioco semplice. Ma è qui che risiedono i punti di forza di DK (lo abbreviamo per comodità!).



Questa sua semplicità concede al gioco di raggiungere un equilibrio tra rilassante e coinvolgente perfetto! È come se il gioco distillasse divertimento in una forma pura e senza restrizioni. È incredibile la facilità in cui mi sono ritrovato a frugrare in mezzo ai

GIUDIZIO FINALE

» Giocabilità 95%
Si fa tutto con pad direzionale e due tasti e lo si fa con soddisfazione.

» Longevità 90%
Bilanciato e strutturato. Porta ad un Nirvana assoluto per chi cerca qualcosa che dia soddisfazione piena.

sotterranei di gioco, girovagando alla ricerca di attività per occupare il tempo. Gran parte del divertimento dipende dalla concentrazione del design; come ogni elemento serve ad offrire divertimento al giocatore velocemente. Colpire il nemico, ad esempio, conferisce una forza tangibile. Il feedback immediato è evidente. I livelli sono bilanciati. Le stanze non sono mai più grandi di un singolo schermo (ricorda la semplicità del Zelda per NES) e i nemici di solito muoiono con pochi colpi ben assestati. Ognuna di queste scelte progettuali crea un gioco costruito attorno a porzioni di tempo in game molto piccole, lasciando poco tra te e il tuo divertimento.

Il gioco fa questo. Utilizza il tempo per strutturare l'esperienza; dare ad un giocatore qualcosa su cui lavorare in modo che sia divertente. Intento encomiabile, che sembra ovvio ma spesso non lo è.

Giocare a questo titolo è divertente. Dannatamente divertente.

C'è un'estetica d'azione perfetta che si bilancia con un livello di difficoltà graduale e che appaga.

È un titolo che dona gioia a chi cerca qualcosa di semplice ed intelligente da scoprire in mezzo a cervellotici roguelike moderni o mostruose avventure tutte tutorial.

Dicing Knight è perfettamente consapevole di che tipo di gioco vuole essere: un dungeon crawler veloce e orientato all'azione che appaghi i giocatori. Difficile pensare a un gioco migliore su WonderSwan, ma anche su altri sistemi.

Riscopritelo!

di **Roberto Del Mar Pirazzini**

NEW GAME

GENESIS
DAWN OF A NEW DAY

Anno: 2010/2024
Editore/Sviluppatore: Retroworks
Genere: Shoot em up
Piattaforma: ZX Spectrum
Sito web: https://retroworks.itch.io/genesis-dawn-of-a-new-day

Nessuno sulla terra aveva mai preso sul serio la minaccia dei terribili alieni Dork (beh, con quel nome... ndNith). Nessuna pensava potessero attaccare... ma l'umanità non impara mai.

La terra è invasa da queste bislacche creature e un solo pilota potrà liberarla e salvare tutti.

È scontato che questo pilota lo guideremo noi come è scontato che per liberarci dai Dork non sarà affatto facile: prima dovremo recuperare tutti i pezzi dell'astronave Genesis e poi affrontarli.

Non è scontato invece recuperare i pezzi, dislocati in cinque pianeti diversi. Una volta assemblati ci si recherà belli e combattivi verso la terra, pronti a liberarla dai Dork.

Uno shoot em up a scorrimento orizzontale con tutti gli elementi dei classici del passato.

Power up, doppio tiro, smart bomb e tanti, tantissimi avversari a riempirci la giornata di sano blastaggio spaziale.

È possibile giocarlo con joystick Kempston o Sinclair oppure coi comandi della tastiera.

Non è un gioco facile, tutt'altro! Possiede un alto tasso di mortalità e spesso il modo migliore per affrontare il livello è quello di eliminare solo i nemici che ci troveremo direttamente di fronte o di approntare alcuni percorsi alternativi.

Insistere, perseverare e combattere... fare un game over e ricominciare... questo è l'imperativo di Genesis.

GIUDIZIO FINALE

» Giocabilità 85%
Power up, doppio tipo di sparo, smart bomb… ricche wave di nemici e un bel level design.

» Longevità 70%
Avrei reso il gioco più “umano”. Sembra partorito dalla mente del Marchese De Sade.

Il gioco scorre fluido e graficamente è piuttosto vario e divertente.

Ha subito una lunghissima gestazione ma è finalmente disponibile sia in formato TAP che in edizione fisica.

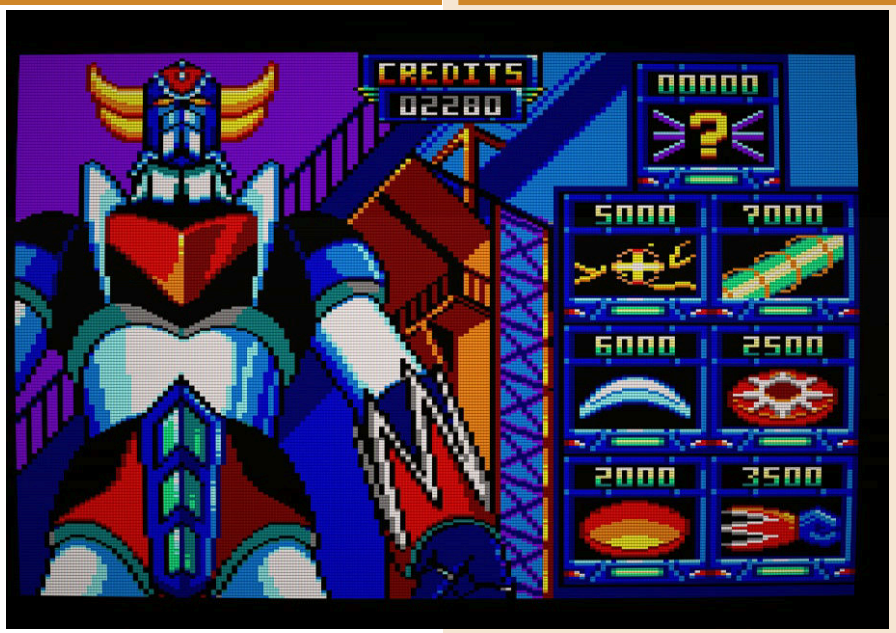
Un buon prodotto per gli amanti del genere e per i masochisti spaziali che cercano un titolo dal forte impatto sui propri nervi.

di **Giampaolo Moraschi**

NEW GAME

GOLDORAK

Anno: 2024
Genere: Shoot em up
Piattaforma: Amstrad GX4000/ Plus
Sito web: https://amstradmuseum.emu-france.info/Fichiers/Projets/Goldorak_CPC/Goldorak_CPC.html



"Va, distruggi il male, va! (Goldrake)! Mille armi tu hai non arrenderti mai, perché il bene tu sei, sei con noi."

Dite la verità, la stavate cantando vero? La sigla scritta da Luigi Albertelli, Vince Tempera e Massimo Luca Bandini e cantata da Actarus al secolo Michael Tadini.

Pezzo quasi mitico del 1978 e sigla finale del cartone Atlas Ufo Robot.

Quanti ricordi suscita Goldrake. Primo cartone "mainstream" lanciato nel 78 e capolavoro ideato da quel genio di Go Nagai.

Terzo nella saga di Mazinga (Z, Grande Mazinga e Goldrake) e figlio di una serie di errori atomici e tipicamente italiani dell'adattamento.

Nato in Giappone con il nome di Ufo Robot Grendizer, arrivò in Europa attraverso la Pictural Film di Jacques Canestrier, che ne aveva acquisto i diritti e cambiato il nome in Goldorak (edizione francese).

Il titolo Atlas Ufo Robot deriva dalla brochure che la signora Nicoletta Artom, manager della rai per i prodotti per l'infanzia, aveva visionato.

Atlas non era un nome altisonante ed esotico ma solo la guida/brochure, e UFO ROBOT il titolo di presentazione e venne frainteso come titolo per la serie.

Col tempo si è cercato di affermare che "Atlas" nel titolo non fu un errore, ma una scelta deliberata dovuta al gradevole (ma casuale) riferimento ad Atlante.

Misteri italici, miti, forse ignoranza del tempo... fatto sta che tutto questo divenne mito.

Chi legge fa parte di quella generazione "Goldrake" ed è sempre contento di vedere giochi o nuove produzioni sul robottone guidato da Duke Fleed.

In attesa della nuova serie Nippo-araba e dopo aver visto il terrificante "Goldrake il banchetto dei lupi" per le nuove console, eccomi su Amstrad per parlare di questa versione nuova di zecca.

Si tratta di uno shoot em up a scorrimento verticale che ci ricorda alla lontana il classico arcade

"Mazinger Z Arcade" di Banpresto.
Scopo semplicissimo. Ci si getta con il Goldrake al recupero dei pezzi del disco di Alcor (Koji Kabuto) distrutti e nascosti dalla flotta di Vega (troppo stressati sti vegani… mancanza di B12).
Ovviamente recuperare i pezzi del disco volante non è impresa facile visto che gli abitanti di Vega sono piuttosto aggressivi e pronti a far la pelle al Robot cornuto.
Per difenderci avremo a disposizione l'armamento basico chiamato Missile Gamma, ma possiamo potenziare il nostro robot upgradando il comparto difensivo ogni volta che rientreremo alla base.
Le armi disponibili sono tutte ben caratterizzate e con funzioni più o meno decisive in alcuni scontri.
Il nostro robot parte con tre vite ed un livello d'energia che viene consumato ad ogni collisione o colpo ricevuto.
Arrivato a zero vite e zero energia si arriva al game over.
Che dire, un classicone del genere con le classiche wave di nemici in caduta dal basso verso l'alto.
La grafica è stata realizzata completamente in Multipaint, un programma davvero ben fatto che consigliamo di andare a visionare a questo indirizzo: http://multipaint.kameli.net/.
Il gioco è molto colorato e ben caratterizzato e, anche se non presenta chissà che quantità di roba sullo schermo, piuttosto veloce.
I nemici sono ben distinguibili e con boss discretamente vari.
Bella la intro realizzata da Eric Cubizolle (TITAN) che si è occupato di tutto l'aspetto grafico e della sceneggiatura.
Non male l'accompagnamento musicale di Pulsophonic. Riprodotte le sigle originali e interessanti gli effetti sonori.
Il gioco gira sulla sfortunata GX4000 oppure su qualsiasi CPC Plus ed in emulazione.
Per gli appassionati del mega robottone di Nagai è un bel prodotto, ma forse un po' noioso. Personalmente mi sarei spinto un po' di più nel numero di nemici e nella loro varietà sullo schermo.
Il gioco è sicuramente molto interessante e da provare.
Da quello che abbiamo potuto vedere in futuro arriveranno anche versione per Atari8bit e altre piattaforme.
È un prodotto curato con tanto di manuale e sito web con tutta la documentazione su sviluppo, musica, grafica e tanto altro.
Se amate questo genere, i robottoni, Go Nagai e siete della generazione Goldrake non fatevelo scappare.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

» Giocabilità 80%
Semplice da giocare e con un buon design dei livelli. Non troppo varie le wave e il tipo di avversari ma interessante. Curato ogni aspetto (grafica, sonoro e presentazione).

» Longevità 70%
Non è difficile e purtroppo non molto vario.

NEW GAME

CECCONOID

Anno: 2024
Editore/Sviluppatore: Thalamus/Triple Eh?
Genere: Shoot em up
Piattaforma: Amiga
Sito web: https://thalamusdigital.itch.io/cecconoid-amiga

Cecconoid è un tipico arcade game. Competitivo, rigoroso, difficile... Un moderno titolo che potrebbe benissimo provenire da una dimensiona parallela dove siamo in piena Golden Age dei videogiochi.

Ricorda Cybernoid, ma con un appeal da moderno roguelike e un pizzico di Metroid.

La storia affascina quanto la sua grafica in bianco, nero e rosso. Trasuda di vibrazioni anni 80.

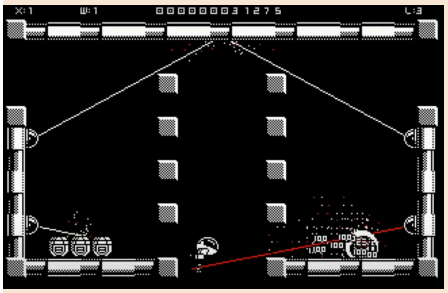
La navicella spaziale Equinox e sotto attacco. Stormlord e i suoi scagnozzi stanno distruggendo tutto quello che è contenuto al suo interno e l'unica speranza per l'umanità è la capsula d'attacco Samurai-1 ovvero noi.

La missione prevede il recupero della chiave del Capitano Solomon e l'apertura del nucleo di Stormlord ovvero il suo annientamento.

Sembra divertente vero? Si, forse la storia è banalotta, ma il risultato è sicuramente divertente.

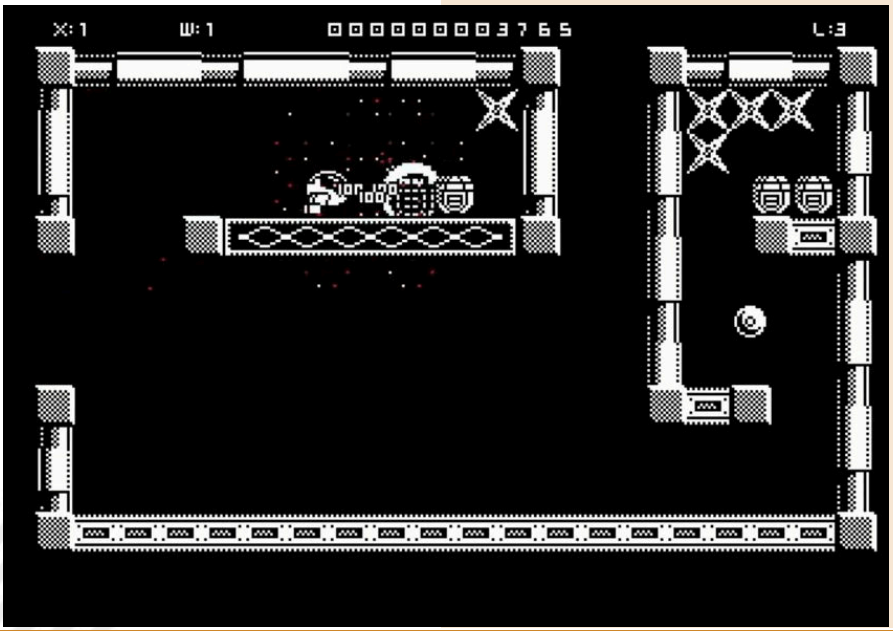
Il gioco è uno sparatutto con scorrimento laterale multidirezionale che non ha un ritmo frenetico. Richiede attenzione nei combattimenti e nei posizionamenti. Ci si concentrerà su tattiche precise per affrontare i nemici con la nostra pixelosa astronave. Nemici decisamente agguerriti e pronti a farci la pelle. Non solo altre astronavi, ma anche missili, mine, trappole, raggi laser... insomma qualsiasi cosa che si muove sullo schermo vuole farci la festa.

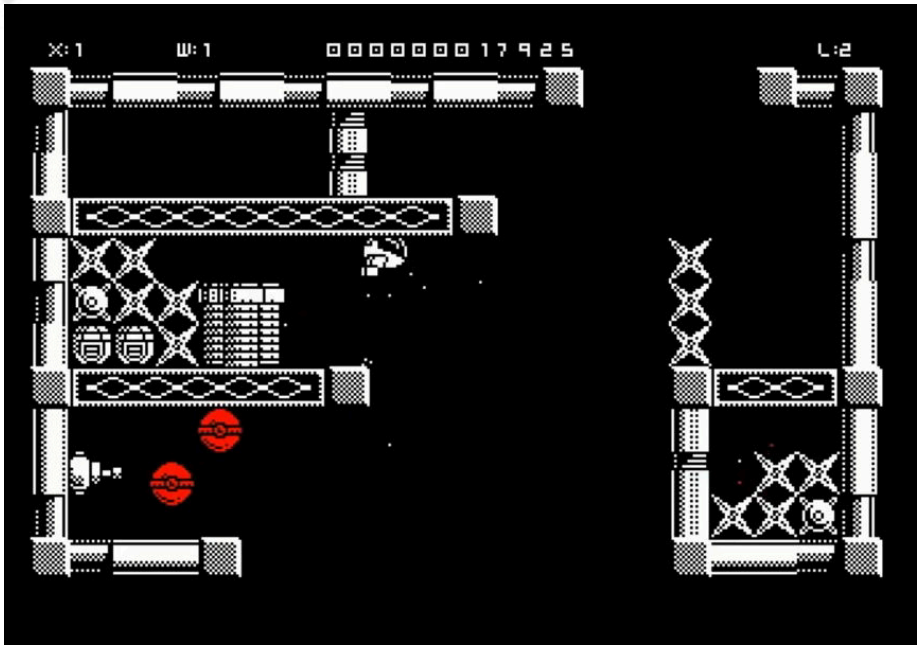
Il gioco fa uso della doppia levetta: il primo joystick serve per muoversi, il secondo per direzionare lo sparo. Nel menù iniziale è configurabile come meglio vogliamo e in base alle nostre esigenze.



Le meccaniche di gioco sono decisamente particolari e fanno fronte anche ad una difficoltà davvero elevata, adatta al giocatore esperto e non al casual gamers.

Esteticamente è un grandissimo prodotto. La grafica in bianco e nero





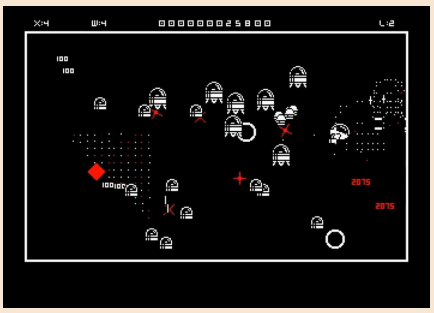
(e rosso) è un gioiello. Un connubio tra vecchio e nuovo con notevoli effetti animati.
Anche il comparto audio è incredibile. Colonna sonora azzeccatissima.
Il titolo al suo interno presenta anche il gioco Eugatron. Una versione ancora più "estrema" del gioco principale.
Un deathmatch game dove si dovranno affrontare orde su orde di avversari sullo stile dei classici come Robotron 2084 o Smash TV.
Bello, fragoroso, distruttivo!
Per gli appassionati di questo tipo di giochi Cecconoid è un'esperienza fenomenale. Alta difficoltà, bella grafica, grandissimo sonoro e un buon level design. Forse adatto ai giocatori più "esperti", ma sicuramente un gran bel titolo.

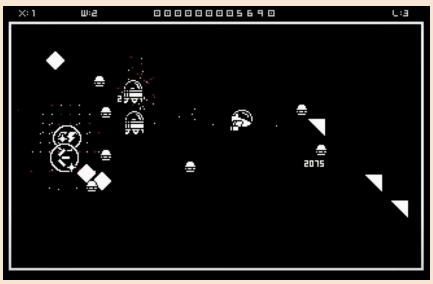
di **Carlo Nithaiah Del Mar Pirazzini**

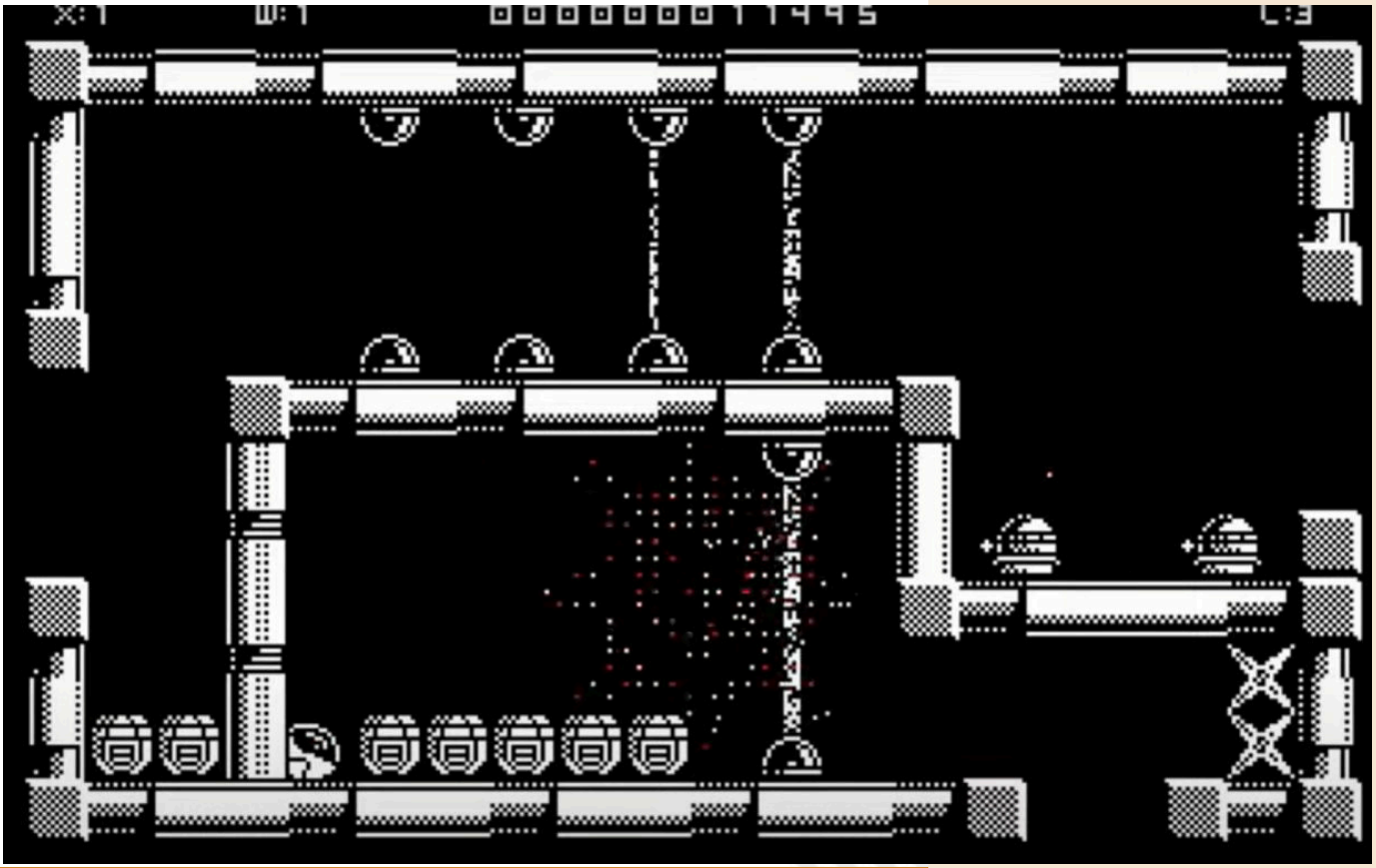
GIUDIZIO FINALE

» Giocabilità 90%
La meccanica delle due leve è interessante e stimolante. Eugatron è un grande valore aggiunto.

» Longevità 85%
Non è un titolo semplice e richiede tanta manualità.







NEW GAME

MIKIE

Anno: 2024
Sviluppatore: Krzysztof "Vega" Góra, Michał "Miker" Szpilowski e Krzysztof "Kaz" Ziembik
Genere: Action game
Piattaforma: Atari 8bit

Mikie era un videogioco arcade del 1984 realizzato da Konami dallo stile irriverente per l'epoca.
Ci portava nel mondo stereotipato delle "High School" americane nei panni di un buffo ragazzetto con pettinatura pompadour e dai modi "grezzi" con poca voglia di studiare e tanta voglia di stare assieme alla sua ragazza che si trovava all'esterno della scuola.
Il gioco si sviluppava in cinque scenari:
Il primo durante la lezione dove il biondo "tamarro" faceva cadere i compagni e le compagne dalle sedie e raccoglieva dei cuoricini sotto queste ultime, mentre il professore cercava di fermarlo lanciandogli, qualche volta, la dentiera.
Il secondo ci vedeva nello spogliatoio alle prese con il professore, un rozzo bidello e il cuoco. Anche qui dobbiamo scappare.
Il terzo era la mensa dove bisognava evitare i cuochi e il sempre più furente professore.
Il quarto era la palestra e in fine si arrivava al cortile dove, evitando grossi giocatori di football americano, si poteva raggiungere l'amata.
Solo tre vite e tanta, tantissima manualità per portare a termine la storia.
Il gioco fu convertito inizialmente per Sega SG 1000 e successivamente per Amstrad CPC, Commodore 64, ZX Spectrum, Acorn Electron e BBC Micro... quindi su tutti i sistemi ad 8 bit dell'epoca...

Mancava giusto una versione Atari 8bit e finalmente è arrivata.
Il titolo è stato realizzato dal gruppo formato da Krzysztof "Vega" Góra, Michał "Miker" Szpilowski e Krzysztof "Kaz" Ziembik, gira su Atari con 128k in formato PAL, il che significa

GIUDIZIO FINALE

» Giocabilità 90%

Mantiene le caratteristiche del coin op ed è realizzato in modo impeccabile.
Comandi perfetti.

» Longevità 80%

Una partita tira l'altra anche se vi farà "uscire dai gangheri" molte volte.

macchine come il 130XE e macchine dotate di espansione di memoria oppure su Altirra in emulazione (configurato a dovere).

E dobbiamo dira che gira molto bene e risulta divertente come un tempo.

Utilizza la modalità grafica Antic 4, ovvero caratteri 40x24 in cinque colori.

Si muove veloce ed è bello da vedere.

Il comparto sonoro è bello. Sono presenti i celebri brani che caratterizzavano il coin op ovvero A Hard Day's Night e Twist and Shout dei Beatles.

La giocabilità è figlia dei suoi tempi. È un gioco del 1984, un mangia gettoni e non poche volte vi fara "impazzire". Non sarà impresa facile superare professori, bidelli, ballerine, insegnati di educazioni e vari energumeni presenti; ma come per il coin op la voglia di ricominciare anche dopo "morti" è devastante. Una partita tira l'altra.

Un altro bel titolo per questa nuova era dei nostri amati retro sistemi.

di **Carlo Nithaiah Del Mar Pirazzini**

Sito web: https://atarionline.pl/v01/index.php?ct=kazip&sub=M&title=+Mikie#kaz_c559e7d8dc286f05c79af72fa972060f

NEW GAME

ROGUECRAFT

Anno: 2024
Editore/Sviluppatore: Thalamus, BadgerPunch Games
Genere: Dungeon Crawler/ Roguelike
Piattaforma: Amiga
Sito web: https://thalamusdigital.itch.io/roguecraft

Ho sempre apprezzato i Dungeon Crawler. Un genere che presenta alcune caratteristiche derivate direttamente dai giochi di ruolo ma che li semplifica portandoli all'interno di labirinti dove conta poco il dialogo parlato ma è più necessaria l'azione. Uno dei primi videogiochi del genere fu Pedit5, realizzato nel 75 da Rusty Rutheford. Un titolo basico che includeva diverse caratteristiche di D&D e la presenza di tanti tesori all'interno dell'avventura.

Tra i più famosi e amati (soprattutto su Amiga) troviamo il mai troppo lodato Dungeon Master, la serie Eye of the Beholder oppure i vari Ultima, Diablo e compagnia bella.

E proprio su Amiga ritroviamo questo genere con Roguecraft, titolo sviluppato dai BadgerPunch Games e distribuito da Thallion, che ci porta all'interno del pericoloso mondo di Mordecom.

Saremo alla guida di Zendar. Uno degli ultimi veri eroi della sua epoca. Si getterà alla ricerca di un potente amuleto nascosto nei sotterranei maledetti.

Prima di farlo è però necessario scegliere la classe di appartenenza del nostro Zendar.

Possiamo decidere tra Guerriero (modalità facile) resistente e fornito di spada, il Ladro (difficoltà media) più leggero, armato di doppio coltello e di una capacità di teletrasporto rapido, ed infine il Mago (difficoltà massima), debole fisicamente ma con la capacità di lanciare potenti proiettili magici.

Ogni classe di gioco ha un proprio indicatore di saluta massimo mostrato in alto con dei piccoli cuori rossi e un indicatore di forza che si trova appena sotto con una barra blu.

È possibile aumentare sia la salute che il livello di forza attraverso

l'uccisione di un certo numero di nemici oppure recuperando alcuni oggetti o pozioni. L'uccisione di un numero cospicuo di avversari rilascerà nella stanza una gemma rossa (aumenterà i cuori) oppure una gemma blu (accrescerà la forza).

Zendar si muove attraverso stanze che verranno rivelate su una mappa laterale ma mano che verranno esplorate.
Lo scopo è trovare la scala che porta al livello successivo. Compito non semplice visto che la chiave è solitamente nascosta da qualche parte nello stesso dungeon e ben protetta.
Una volta recuperata, la porta del nuovo livello si apre e viene contrassegnata sulla mappa da un pallino blu.
I dungeon di Roguecraft vengono generati in modo procedurale e presentano il classico menù di questo genere di giochi: tesori, segreti, trappole e una vagonata di nemici (FATE ATTENZIONE AI MIMIC TRAVESTITI DA FORZIERI!).
Il proseguire dell'avventura non solo ci permetterà di potenziarci ma presenterà anche nemici via via più coriacei, consigliamo quindi di bilanciare saggiamente la forza prima di affrontare i livelli più in basso.
Il movimento di gioco è a turni, con Zendar che si muove facendo il suo turno e i mostri nella stanza che si muovono in risposta alla sua azione.
Si ingaggia il combattimento spostandosi in direzione del mostro di turno e affrontandolo nel quadrato accanto (o lanciando da lontano un proiettile magico nel caso della scelta del Mago).
In automatico avverrà lo scontro.
L'uso del pulsante di fuoco è per l'inventario. Premendolo si passa ad un pannello situato in basso dove si potrà selezionare e leggere quello che abbiamo raccolto nel nostro girovagare.
Solitamente si tratta di pozioni e ce ne

GIUDIZIO FINALE

» Giocabilità 95%
Bellissimi livelli generati proceduralmente e un ottimo sistema di gioco. I Dungeon sono sempre vari e ben fatti.

» Longevità 91%
Le tre classi di gioco offrono esperienze diverse d'approccio ai dungeon. Giocare con il mago è un vero inferno ma il titolo non stanca mai.

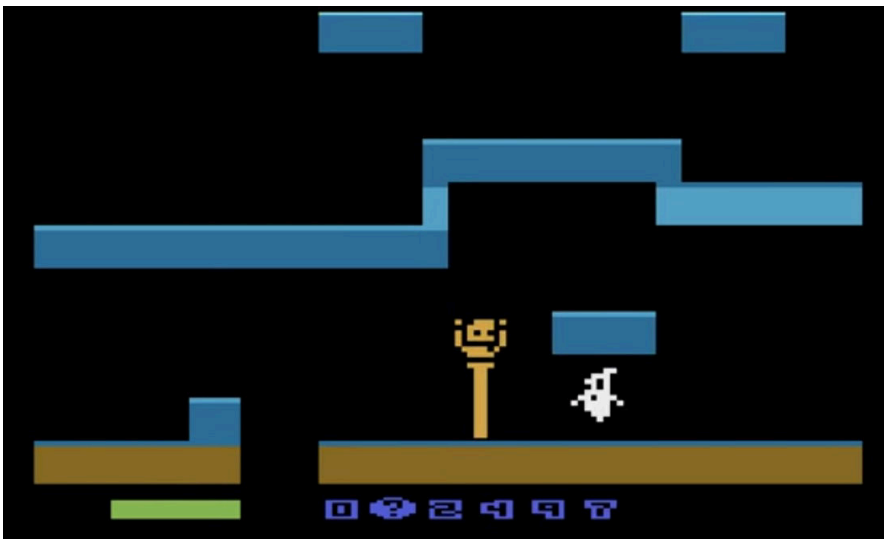
sono diverse: la pozione verde è un classico cura ferite, la pozione blu e quella rossa invece cambiano effetto in ogni partita.
Tra i possibili effetti possiamo citare il Congelamento, l'Esplosione, Rivelazione e il Teletrasporto. È presente anche una folle bottiglia contenente Alcol che metterà a dura prova i nervi dei giocatori più esperti con effetti curiosi.
Se si dovesse morire prima di arrivare al mega boss finale, apparirà una schermata di Game Over che riassume tutto il punteggio, il livello raggiunto, quanti forzieri recuperati, quanti segreti rivelati e... quale mostro ci ha ucciso!
Lo sviluppo di Roguecraft è stato piuttosto lungo e molto atteso. Mettendo le mani su questa versione per la stampa possiamo dire con certezza che ne è valsa la pena perché Roguecraft è un vero gioiellino. È bello da vedere e sentire. Una piccola opera d'arte in pixel che stupisce per i dettagli e per la meravigliosa scelta cromatica.
Il dettaglio per ogni singola cosa è maniacale. Un esempio? Le animazioni dei mostri. Meravigliose.
C'è tantissima varietà nell'affrontare i livelli con le tre classi di gioco, c'è varietà nei dungeon e nei mostri.
E che dire della colonna sonora... un bomba che coinvolge e crea un'atmosfera unica e seducente.
Il punto di forza è però nel design dei livelli e nel modo in cui si viene coinvolti nell'esplorazione. Se all'inizio appare come un semplice gioco d'esplorazione col susseguirsi delle partite diventa una dipendenza. Unico piccolissimo appunto è per quanto riguarda la difficoltà nella modalità "Mago". Effettivamente il livello di sfida con questa classe è quasi impossibile. Da veri eroi. Un'inezia, ma è consigliato fare tantissima pratica con le altre classi prima di imbarcarsi nell'avventura con il maghetto. Roguecraft può essere giocato su Amiga con chipset ECS/AGA, 1 Mb di Chip ram e 1 Mb di altra ram e processore 68000. I ragazzi di BadgerPunch consigliano una configurazione A1200 per aver miglior esperienza di gioco e meno problematiche di rallentamento. Ovviamente funziona senza problemi su qualsiasi emulatore e su A500 mini.

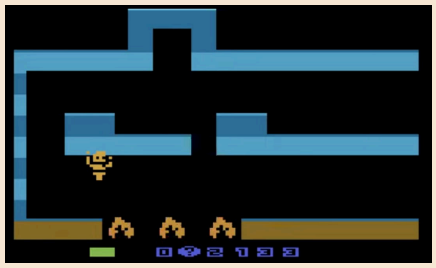
di **Carlo Nithaiah Del Mar Pirazzini**

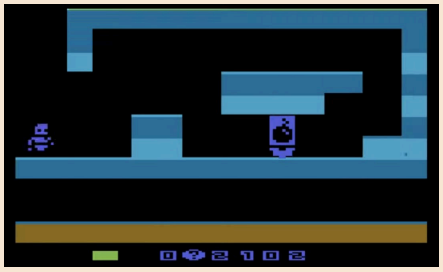
NEW GAME

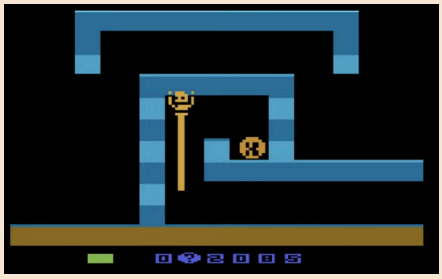
ROBO TITO

Anno: 2024
Editore/Sviluppatore: VHZC Games
Genere: Platform/Action
Piattaforma: Atari 2600
Sito web: https://vhzc.itch.io/robotito









Il piccolo robot Tito ha un solo difetto. La sua programmazione non è andata a buon fine e soffre di una terribile "allergia" alle cose mostruose, ai fantasmi e ai mostri.

Per sua sfortuna si trova intrappolato in un posto terrificante infestato da tutte le creature più orribili del creato. Però il posto è sui nostri Atari VCS ed il gioco è davvero divertente nella sua semplicità.

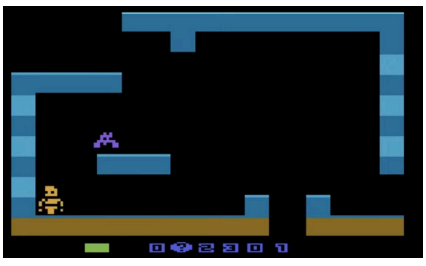
Tito non è armato di nessun raggio laser o super forza, ma si può allungare e aggrappare alle pareti in alto per sfuggire a trappole e mostri.
Per noi è semplice! Dobbiamo aiutarlo ad uscire da lì.

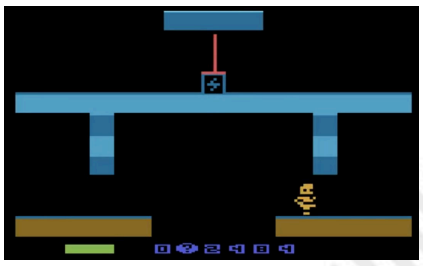
Il gioco è semplice e simpatico e devo ammettere di essermi divertita a giocarci sul VCS che mi hanno prestato. È una bella sfida quella di portare Tito in salvo e non richiede chissà che tipo di "cervellotico" sistema. Basta scappare e comprendere al meglio i movimenti dei nemici.

Un gioco semplice, programmato con cura e molto divertente.
Mi sono piaciuti tantissimo gli sprite di Tito e di alcuni mostri (adoro il fantasma).

Il gioco è veloce e si lascia amare.
Posso aggiungere altro? Direi di no se non quello di supportare lo sviluppatore e di caricare il gioco su VCS o in emulazione.

di **Marta Rossmann**





GIUDIZIO FINALE

» Giocabilità 90%
Mi sono piaciute molto le meccaniche e il design dei livelli di gioco. Adoro lo sprite.

» Longevità 80%
Una sfida bilanciata. Non troppo difficile né troppo semplice. Molto godibile.

NEW GAME

THE HEART OF SALAMANDERLAND

Anno: 2024
Editore/Sviluppatore: Juan J. Martinez
Genere: Platform
Piattaforma: Amstrad CPC/ GX4000
Sito web: https://www.usebox.net/jjm/heart-of-salamanderland/

Patton è il più grande avventuriero del suo tempo. Sempre pronto ad imbarcarsi in avventure spericolate; tra Templi Maledetti, Arche perdute o misteriosi cimeli templari. Sempre pronto a srotolare la sua frusta… Ehi! Ma sbaglio o ci ricorda Indiana Jones? The Heart of Salamanderland è un platform game per Amstrad CPC (dai 64k di Ram in su) ed il protagonista è chiaramente ispirato al celebre eroe di Lucas.
Lo scopo è recuperare una reliquia nascosta lungo 55 stanze belle ricche di mostri e trappole. La frusta del nostro Patton sarà la nostra unica arma di difesa/offesa e, per arrivare alla reliquia sacra, dovremo esplorare i sotterranei e recuperare le 9 lacrime del guardiano. A quel punto uccidere il mega boss finale e reclamare il meritato premio.

La struttura ricorda quella dei più rodati platform dell'epoca d'oro. I livelli sono ben congeniati e in generale non troppo impegnativi o cervellotici, ma strutturati con una curva di difficoltà crescente.
Il titolo è stato sviluppato da Juan J. Martines che si è occupato di grafica, suono e codice. Un lavoro ben fatto perché tutto risulta molto colorato, animato bene e con un buon accompagnamento musicale.
C'è anche lo zampino di Eric Cubizolle, in arte TITAN, che ha realizzato la schermata introduttiva (molto bella) ispirandosi ai classici come Castlevania, Rick Dungerous e Montezuma's Revenge.

Il titolo supporta i due pulsanti: uno per la frusta ed uno per il salto.
Ed è proprio saltare la chiave di volta. The Heart of Salamanderland appartiene a quel genere di giochi dove la precisione da piattaforma a piattaform è la chiave di volta per vedere i titoli di coda.
I salti devono essere "perfetti".

Il titolo è disponibile per il download gratuito su usebox.net che vi ho linkato nel box apposito. Gli autori stanno

GIUDIZIO FINALE

» Giocabilità 80%
I due pulsanti di gioco sono essenziali. Saltare con il solo joystick può portare ad esaurimento. I livelli di gioco sono ben strutturati.

» Longevità 75%
55 schermi di gioco possono sembrare tanti ma in realtà, con un po' di pratica il compito si porta a casa facilmente.

realizzando anche un'edizione fisica prevista in uscita per fine 2024 sul sito di Poly Play.

Ma alla fine è un bel gioco? Si, un bel platform che si giochicchia volentieri e che si ricarica tutte quelle volte che vogliamo senza la pretesa di essere un capolavoro assoluto.

Sufficienza piena.

di **Marta Rossmann**

NEW GAME

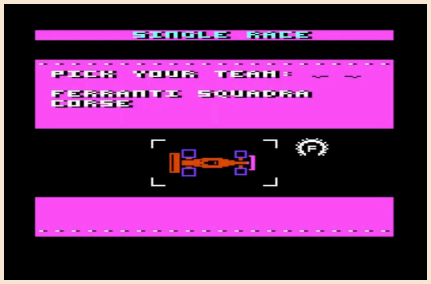
FORMULA V20 1985

Anno: 2024

Editore/Sviluppatore: Huffeduff

Genere: Driving

Piattaforma: VIC 20 (+24kb di espansione)



Formula V20 1985 è un titolo di corse per VIC 20 che omaggia i grandi classici come Pit Stop II e Pole Position. Nasce per essere giocato su un VIC espanso a 24k.

Il titolo è realizzato in Kick Assembler, strumento versatile utilizzato anche per altri titoli dello sviluppatore.

E' possibile selezionare sette diversi team e gareggiare su quattro circuiti diversi. Si può scegliere la modalità campionato oppure correre in una singola gara selezionandola così come è possibile scegliere il livello di difficoltà.

Il gioco è molto veloce e si muove bene. Grafica convincente abbastanza dettagliata e effetti sonori adeguati al genere.

Si gioca abbastanza bene con l'ausilio dal pad/stick e il livello di difficoltà è livellato sulle piste partendo da un livello basso fino al massimo della sfida possibile.

Siamo di fronte ad un buon prodotto per VIC, adatto agli appassionati del genere e agli amanti di Pit Stop 2 e soci. E' possibile eseguirlo su Vice in emulazione selezionando nelle impostazioni le opzioni per l'espansione di memoria.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

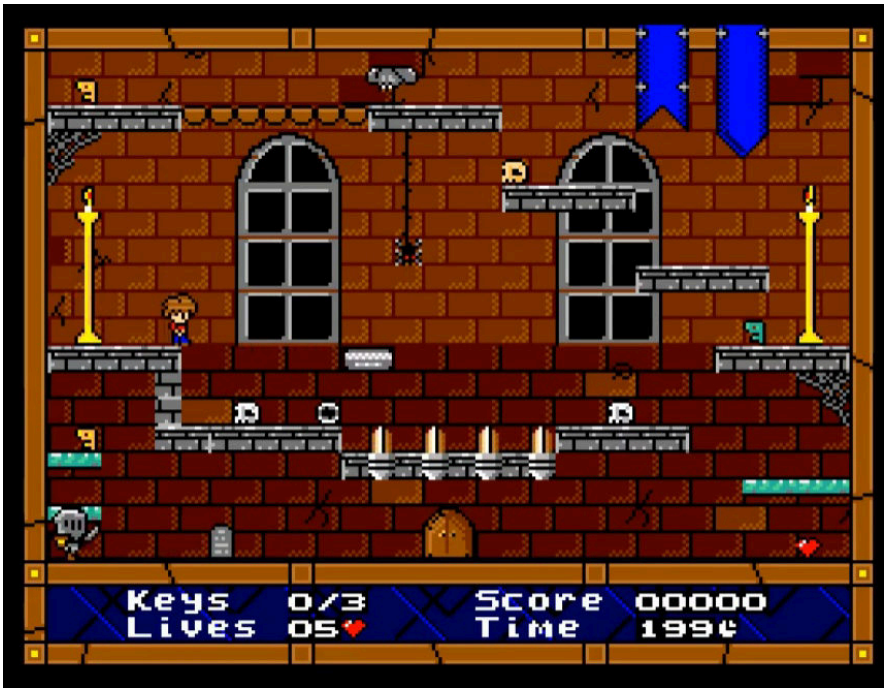
» Giocabilità 75%
Un buon gioco di corse per VIC espanso sulla falsariga di titoli storici.

» Longevità 70%
Avrei voluto più piste di gioco.

NEW GAME

PRINCESS PALOMA'S RESCUE

Anno: 2024
Editore/Sviluppatore: InfiniteMSX
Genere: Platform/Puzzle
Piattaforma: MSX2 (64kb/128kbvm, MSX-Music)
Sito web: https://www.msxdev.org/2024/05/03/msxdev24-01-princess-palomas-rescue/



Il primo titolo della bellissima competition MSXdev24 è questo Princess Paloma's Rescue, un gioco d'azione per MSX2 che vede un intrepido eroe imbarcarsi nella ricerca di una principessa rapita.
Adoro l'MSXdev, credo sia una delle migliori competition per nuovi titoli in assoluto e ogni anno ci sforna e mostra piccoli gioielli.
Questo solido gioco di InfiniteMSX si mette in pole position per attirare l'attenzione dei giudici e di noi addetti del settore.
La principessa Paloma è stata rapita da un perfido mago chiamato Morgath, il nostro eroe ha il compito di attraversare cinque regni distinti, recuperare delle chiavi e cercare di arrivare incolume dalla bella Paloma. Impresa come sempre in questo genere di giochi, non semplice. È un titolo ricco di trappole e ostacoli e viene richiesto ingegno e un po' di astuzia e materia grigia per superare i livelli.
Il titolo è nativo MSX2 e richiede funzionalità MSX-Music, standard audio del 1987.
Esteticamente è gradevole sotto tutti i punti di vista. C'è una grafica "pucciosa" e ben animata e una bella musica condita di buoni effetti sonori.
Attenzione! Non è un gioco semplice e mette alla prova le capacità di ragionamento in certe situazioni.
Alcuni quadri sono decisamente ostici da portare a termine.
Però è un gran bel giochino e non ci resta che guardare cos'altro ha in serbo questa edizione 2024 della MSXDev.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

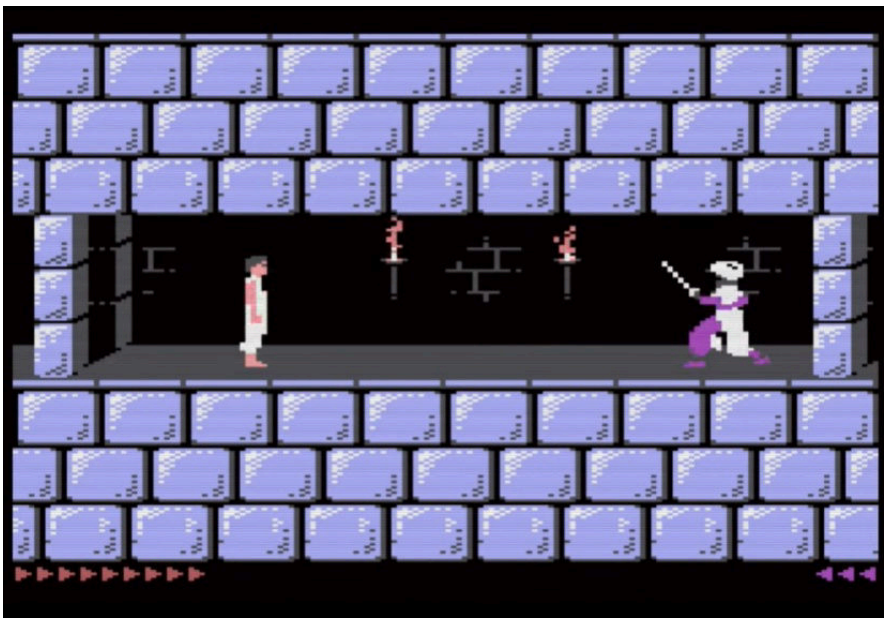
» Giocabilità 85%
Meccaniche semplice e bei livelli di gioco.

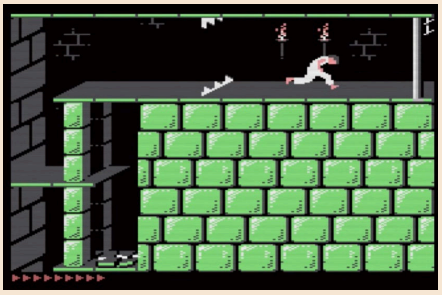
» Longevità 80%
Difficile in alcuni punti e richiede attenzione.

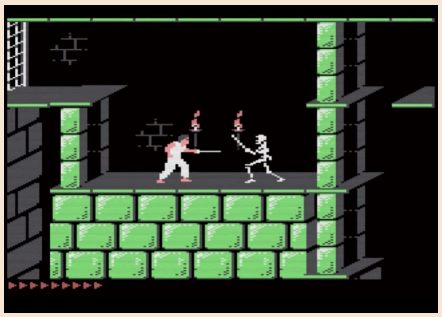
NEW GAME

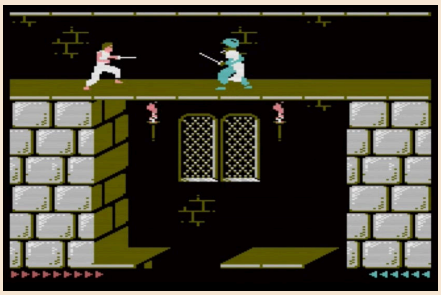
PRINCE OF PERSIA

Anno: 2024
Editore/Sviluppatore: TCFS
Genere: Platform
Piattaforma: Plus 4
Sito web: https://plus4world.powweb.com/software/Prince_of_Persia









W il Plus 4!
Macchina che per il sottoscritto “ha avuto poco tempo”.
Poco tempo per essere conosciuta dal pubblico, poco tempo per essere conosciuta dagli sviluppatori e dai programmatori e, soprattutto, poco tempo per essere presa sul serio dalla stessa Commodore.

Eppure, nel 2024 mostra i muscoli.
Il team composto da TCFS, Unreal e Csabo ci hanno dimostrato in questi anni come un’ottima conoscenza porta sempre a risultati impressionanti.
Prince of Persia per PLUS 4 è una meraviglia!

Coloratissimo e animato con una grande cura. Dettagliato, veloce, con tutti i colori del TED brillanti e vivaci. Unica pecca, il sonoro che rallenta un po’.

Il Plus 4 è un home computer che deve essere riscoperto e apprezzato. Il team di TCFS lo sta facendo.
Ha dato una nuova vita ad una macchina che purtroppo ha avuto davvero pochissimo tempo per farsi voler bene.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

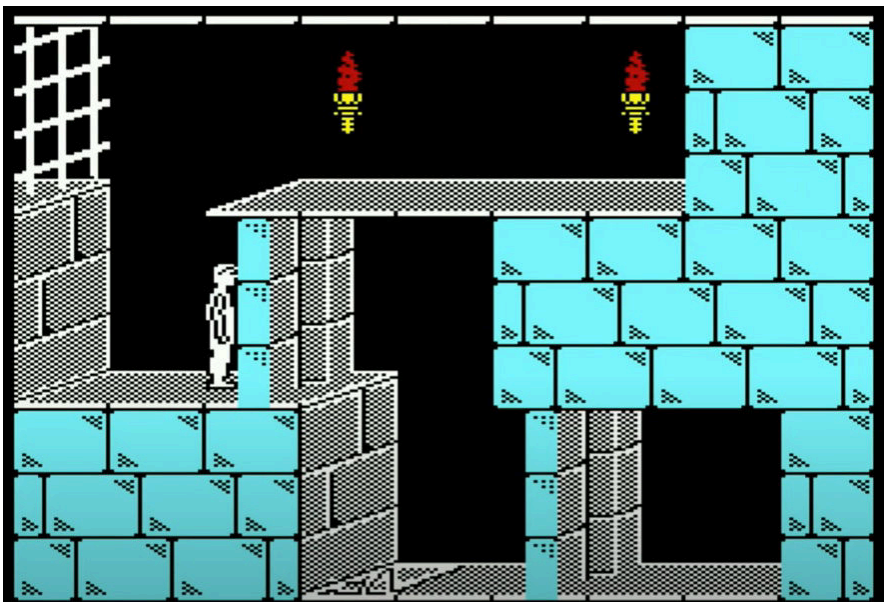
» Giocabilità 90%
Una meraviglia per gli occhi con tutti i livelli e la giocabilità del titolo originale.

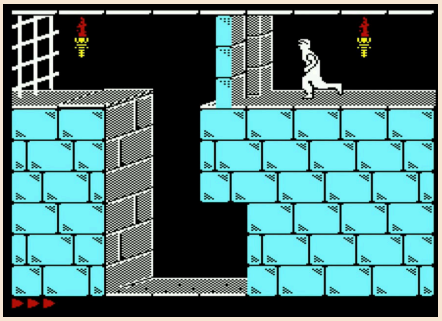
» Longevità 90%
È Prince of Persia… ho detto tutto.

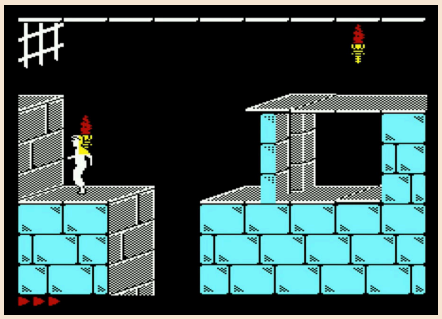
NEW GAME

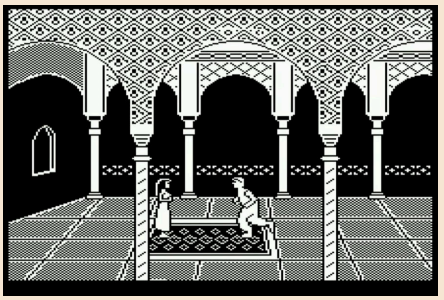
PRINCE OF PERSIA

Anno: 2024
Editore/Sviluppatore: LeChuck
Genere: Platform
Piattaforma: VIC 20 (+35kb)
Sito web: https://sleepingelephant.com/ipw-web/bulletin/bb/viewtopic.php?p=120870#p120870



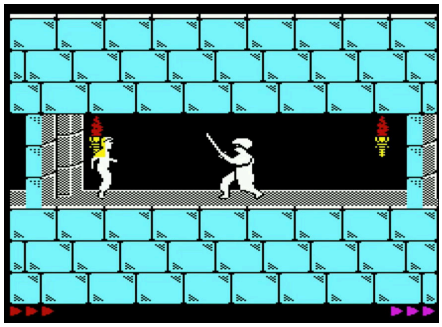


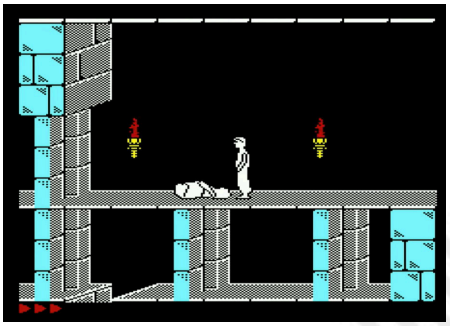




Sono passati 35 anni dall'arrivo del Principe sui nostri schermi di gioco. Una saga iconica con numerosi seguiti e con un folto numero di estimatori.
Il fascino del primo capitolo però è innegabile, lo abbiamo visto in tutte le sue forme ma sembrava impensabile su un VIC 20.
I limiti di memoria del piccolo computer domestico di Commodore sono sempre stati un ostacolo per la realizzazione di certi titoli... Questo fino ad oggi...
Si può fare! E lo ha fatto lo sviluppatore LeChuck, ispirato dalla versione (bellissima) per ZX Spectrum realizzata da Nicodim.
La versione di Prince of Persia per VIC 20 presenta solo 9 livelli (più finale) e diverse mancanze sotto il punto di vista delle animazioni ma è... bellissima!
Per essere giocata occorre un'espansione di memoria da 35Kb e un buon joystick, per il resto la magia di Prince of Persia c'è tutta.
Il gioco utilizza 27 colonne x 29 righe e quindi gira su formati PAL.
Il Principe su VIC si muove bene e si lascia giocare presentando tutte le peculiarità di questo amatissimo titolo.
Ora non ci resta che attendere la versione per Plus 4 in via di sviluppo.

di **Giampaolo Moraschi**





GIUDIZIO FINALE

» Giocabilità 85%
Limitazioni tecniche e meno livelli ma è sempre un grande gioco.

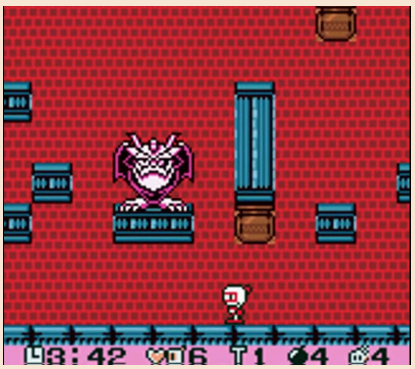
» Longevità 90%
E' Prince of Persia, un titolo che vuole essere giocato fino in fondo.

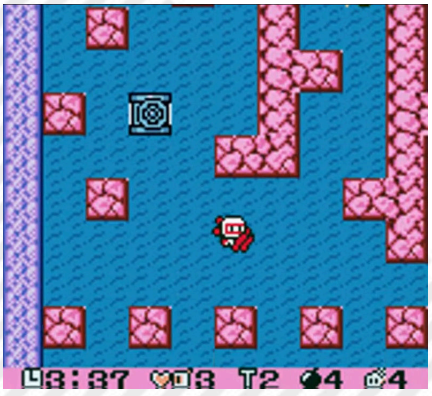
POCKET BOMBERMAN

Anno: 1997 versione Game Boy, 1998 versione Game Boy Color
Editore/Sviluppatore: Nintendo, Hudson Soft
Genere: Platform
Piattaforma: Game Boy, Game Boy Color

Oggi vi parlo di questo piccolo gioiellino che è una vera perla per Game Boy.
Una versione completamente diversa del classico gioco di Bomberman perché a differenza della solita schermata labirintica (infatti si colloca nel genere "Maze" come Pac-Man), questa è un platform dove il nostro protagonista deve attraversare cinque mondi (Forest, Ocean, Wind, Cloud e Evil) e alla fine di ogni livello affrontare un Boss, l'ultimo sarà per l'appunto Evil (che stranamente non hanno censurato) ma in breve la trama: "Molto tempo addietro il Sole fu avvolto da una nuvola oscura e sinistra. Una vecchia leggenda sostiene che quella nube era il risultato di un potente mostro che aveva lanciato una maledizione sulla "Spada del Sole" sigillando il suo potere. L'unica speranza rimasta per far tornare la luce sulla terra è quella di raccogliere le cinque Pietre del Potere, sorvegliate dai mostri malvagi di Evil Mountain. Solo le Pietre del Potere hanno la forza necessaria per rompere definitivamente la maledizione della spada e far tornare le cose alla normalità."
Tra power-up che ci fanno piazzare più bombe, altre che ci fanno esplodere le bombe a nostro piacimento, e tanti altri come quella delle ali che ci fa saltare in alto, raccogliamo le nostre pietre per andare avanti con i livelli e sconfiggere i vari mostriciattoli e nemici che ci si parano davanti in un'avventura unica per Game Boy!
In più per chi avesse una repulsione per il grigio/verde dello schermo esiste la versione Color uscita nel 1998, non contenti però ci sta anche un mini-game la modalità "Jump" dove Bomberman dovrà essere guidato in un percorso in salita e dovrà far esplodere i vari blocchi che si presentano davanti per continuare il percorso.
Un platform carino e anomalo che mescola gli elementi classici di Bomberman a quelli di Super Mario. La grafica è semplice ed essenziale così come il reparto sonoro.
È un titolo da riscoprire, forse non il capolavoro assoluto ma decisamente un bel gioco.

di **Barbara "Morgana" Murgida**





GIUDIZIO FINALE

» Giocabilità 80%
E' un platform classico con alcuni tocchi di genio. Semplice da giocare e imparare.

» Longevità 60%
Purtroppo finisce troppo presto... troppi pochi livelli!

NEW GAME

THE AMIGA ARCADE CLASSIC CD32

Anno: 2024
Editore/Sviluppatore: JOTD666
Genere: Compilation di giochi arcade
Piattaforma: Amiga CD32
Sito web: https://jotd666.itch.io/arcade-classics-for-cd32

Una bella compilation come nella gloriosa Golden Era!
Mi emozionavo sempre quando uscivano questo tipo di compilation o quelle in stile Budget. Era come una grande lotteria.
The Amiga Arcade Classic CD32 è una iso (quindi avviabile da CD32) contenente tutti i titoli sviluppati da JOTD per Amiga partendo da alcuni classici da sala giochi. Porting diretti da Z80 a 68000.
Al suo interno troviamo Bagman e Super Bagman, Karate Champ 2 player, Moon Patrol, Ms. Pacman e Pacman, Amidar, Donkey Kong, Galaxian, Scramble, Pengo, Pheonix, Tetris e Xevious.
Alcuni titoli li abbiamo recensiti nella rivista qualche numero fa. Sono tutti ottimamente portati su Amiga e meritano di essere scoperti e riscoperti. In particolar modo Tetris e Xevious, praticamente perfetti.
Questa compilation è totalmente gratuita e scaricabile dalla pagina Itch.io dello sviluppatore.
Funziona su macchine reali ed in emulazione.
Personalmente avrei curato con maggior attenzione il menu di selezione dei titoli, magari realizzando qualcosa di "graficamente" gradevole.
Ne sentivamo il bisogno? No, ma rimane un buon pacchetto di giochi da provare o far conoscere ai nostri figli/nipoti.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

» Giocabilità 75%
Ci sono tanti titoli del passato con le limitazioni del gameplay del periodo d'appartenenza. Alcuni titoli sono meravigliosi, altri… meno.

» Longevità 80%
C'è tanta "roba" da giocare e da provare.

MARATHON

Anno: 1994
Genere: Sparatutto in prima persona
Musiche: Bungie
Editore: Bungie (originale) Team Aleph One (realizzazione di porting moderni)
Piattaforma: Windows/Mac/Linux

Gli anni novanta sono stati l'epoca d'oro degli sparatutto in prima persona, hanno visto la nascita di titoli importanti come Wolfenstein 3D e Doom, quest'ultimo ha indirettamente coniato il termine "Doom-like", una etichetta che veniva data a qualsiasi FPS uscito da lì in poi (dovrà passare qualche anno prima che il termine First Person Shooter o FPS, divenga il termine ufficiale di questo genere di videogiochi).

Tra quei pochi sviluppatori, per l'epoca, che provarono a cavalcare l'onda del successo di Doom, troviamo l'ormai famosa Bungie, che tento di rinnovare la formula degli sparatutto dell'epoca prima con Pathway into Darkness e subito l'anno successivo con Marathon. Marathon è ambientato nel lontano futuro, interamente all'interno della nave colonia che porta lo stesso nome, abbordata da una fazione composta da varie razze aliene. Il giocatore impersona un addetto alla sicurezza che ha lo scopo di salvare la nave. Attraverso i vari livelli si incontreranno vari terminali, ognuno di essi ci metterà in contatto con tre intelligenze artificiali chiamate Leela, Durandal, and Tycho. Questi terminali, tramite un testo, rileveranno dettagli importanti sulla trama ma anche sugli alieni e le IA stesse, chi sono e quali sono le loro intenzioni o compiti.

La prima particolare distinzione da Doom è proprio il modo in cui viene presentata la trama, che questa volta non è solo una semplice scusa per farsi strada tra i nemici, ma sarà una cosa attiva e importante, stesso discorso vale per i personaggi, sia alleati che nemici, che non saranno semplici comparse.

Un'altra particolarità è la possibilità di guardare In alto e in basso, all'epoca (parliamo del 94) la visuale negli sparatutto alla Doom si muoveva soltanto verso sinistra o destra. È

GIUDIZIO FINALE

» Giocabilità 80%

Marathon presenta dei livelli forse claustrofobici o a volte confusionari, ma ci si accorgerà subito che gli ambienti di gioco lo rendono godibile e meno monotono, ci si troverà dal semplice leggere terminali e cercare chiavi per aprire porte, ad aiutare membri dell'equipaggio in difficoltà o risolvere puzzle per proseguire in una determinata zona.

» Longevità 80%

Il gioco è abbastanza lungo, con mappe grandi e con livelli che spesso possono essere completati in modo non lineare.

presente anche la possibilità di ricaricare le armi, una caratteristica oggi diventata la norma negli sparatutto, anche se è possibile farlo in automatico solo quando finisce il caricatore, le munizioni infatti non saranno abbondanti e si dovrà fare attenzione a non finirle, poiché potranno essere trovate solo in giro per la mappa e in relativamente poche quantità. È anche presente un sensore di movimento, altra novità rispetto a Doom, che permetterà di rilevare la posizione dei nemici.

In classico stile Bungie, la trama gira intorno a temi filosofici che si riscontrano nel punto centrale di quest'ultima, ovvero l'IA che diventa senziente e inizia a provare vari sentimenti, un tema classico della fantascienza, ma questa volta presentato in modo davvero interessante e coinvolgente.

poiché Il codice sorgente venne rilasciato gratuitamente, è possibile scaricare il gioco e i suoi sequel gratuitamente da Steam o dal sito del team che ha lavorato al porting, Aleph One.

di **Maurizio Diamanti**

NEW GAME

KIEN

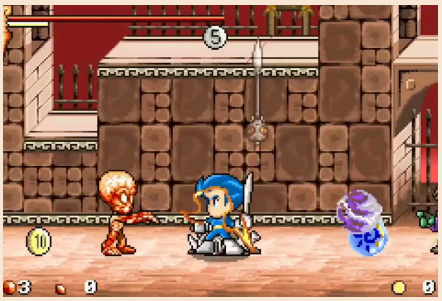
Anno: 2024
Editore/Sviluppatore: Incube8
Genere: Platform/Action
Piattaforma: Game Boy Advance
Sito web: https://incube8games.com/en-eu/products/kien-gba

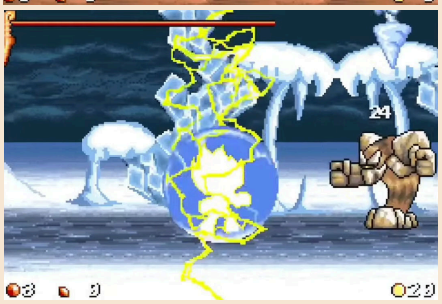
Il gioco più posticipato di sempre è finalmente arrivato sui nostri GBA.
Mi riferisco a Kien, un action game che si potrebbe definire un "Pre-souls" che doveva uscire nei primi anni 2000 proprio su Game Boy Advance.

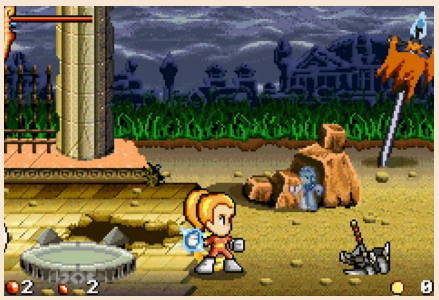
Purtroppo, alcune cause di forza maggiore e notevoli ritardi lo hanno super-posticipato fino ai giorni nostri.
Ma di che si tratta?
È un platform d'azione con diversi elementi presi in prestito dai giochi di ruolo, che prende il giocatore e lo porta attraverso 23 livelli fantasy piuttosto impegnativi e complicati.
Il gameplay ricorda Ghouls 'n Ghost per ritmo di gioco incessante e per la mole di nemici pronti a far la pelle ai protagonisti (ad inizio avventura possiamo decidere se giocare con il guerriero o la maga).
La caratteristica non lineare consente al giocatore di esplorare il vasto mondo nel modo preferito e consentendo di potenziare al meglio il proprio personaggio.
Disponibile in due edizioni: la prima in digitale che contiene la rom di gioco e il manuale in formato pdf, la seconda invece è in formato cartuccia con tanto di scatola, libretto e foglietto pieno di adesivi. Diciamo che il costo della cartuccia è altino.

Ma dopo tutti questi anni d'attesa possiamo ritenerci soddisfatti? Direi non completamente.
Kien è un bel gioco ma presenta alcuni problemi.
La grafica è sicuramente brillante ma infantile, con i nostri eroi e mostri che sembrano presi da una serie di cartoni animati di Cartoon Network. Le immagini statiche che raccontano la storia sono poco dettagliate e "scollate" dal contesto.
Simpatiche le animazioni ma decisamente poche e ripetitive.
Il sonoro è terribile, al limite dell'insopportabile e vi consiglio di abbassarlo il prima possibile.
Lo schema di gioco alla GNG è interessante ma diventa noioso a causa della ripetitività di livelli e nemici.
Scomodo anche il sistema di salvataggio a PASSWORD dove sarà necessario segnare alla fine di ogni livello la parola d'ordine per accedere.
Ma uno slot per salvare? Nel 2024 si poteva anche fare.
Insomma, ci aspettavamo qualcosa di più… molto di più.

di **Marta Rossmann**







GIUDIZIO FINALE

» Giocabilità 40%
Livelli, mostri e schema di gioco ripetitivo.
Il respawn dei nemici è mal gestito e sicuramente esagerato.
Pessimo sonoro.

» Longevità 50%
Non impossibile da terminare ma decisamente noioso.

3D POOL

Anno: 1989

Editore/Sviluppatore: Firebird/ Microprose Software

Genere: Sport

Piattaforma: Commodore 64

L'estate è arrivata e piano piano anche le vacanze; Per molti di noi che hanno vissuto al massimo gli anni 80 e 90, andare nelle località turistiche equivaleva passare intere giornate nelle sale giochi tra decine di cabinati arcade giocando a suon di gettoni e ad osservare gli altri che giocavano, magari quelli più bravi di noi che terminavano il gioco con un solo gettone.

Nelle numerose sale giochi che abbiamo visitato, oltre ai cabinati arcade, c'erano anche tavoli da ping pong, calcio balilla e soprattutto per i più adulti, il biliardo. Non il classico che trovavamo nei baretti di paese, bensì quello all'americana, un po' diverso dal classico a cui eravamo abituati.

Sui computer domestici non tardarano di certo ad arrivare simulazioni di biliardo, così che anche noi più piccoli potevamo già giocare e imparare sin da subito le regole. 3D Pool della Firebird è un buon simulatore 3D che riproduce fedelmente le regole e lo stile di gioco del biliardo americano. Si comincia la partita con le biglie poste a triangolo e la palla bianca pronta ad effettuare il primo tiro, che sarà quella che dovremo utilizzare per mandare tutte le nostre biglie in buca.

La prima biglia ad essere mandata in buca determinerà quale dei due tipi dovremo giocare se saremo noi i primi a fare buca e in gioco ci sono le biglie piene, ossia quelle a tinta unita e le seconde cerchiate, tutte quante numerate più una nera (la numero8) che dovà essere mandata per ultima; Se per errore la manderemo prima, subiremo la sconfitta diretta.

Come tutti i giochi 3D sui computer 8 bit, potremo notare una lentezza che a mio avviso non ho trovato così noiosa come in altri. Le opzioni disponibili sono piuttosto studiate per dare al gioco una bella longevità, dalla partitella singola, la pratica (molto utile) un torneo che inizia dai quarti per poi affrontare in finale Joe Maltese, noto campione di biliardo dell'epoca.

I comandi sono semplici, con il solo tasto fire si deve solo definire la potenza di tiro con una barra di potenza situata sulla schermata in alto e l'effetto della pallina calibrando una crocetta posta sulla biglia bianca accanto all'indicatore di potenza.

Se nel gioco ci fosse stata anche la stecca, sarebbe stato più realistico ma è comunque un buon gioco del tavolo verde che merita di essere vissuto dagli appassionati e non solo. Il livello sonoro è invece inesistente a parte le biglie quando si toccano e vanno in buca, dopotutto la concentrazione è essenziale in un gioco come questo e le musiche nei giochi sportivi durante le gare non mi sono mai state particolarmente simpatiche, nemmeno se fossero state le mie hit preferite.

Spero vi stiate organizzando bene per le vacanze portando sotto l'ombrellone il nuovo numero di questa rivista, qualche gioco a tema estate e vacanze e soprattutto tanti bei ricordi di quegli anni d'oro, dato che ahimè le sale giochi ormai sono del tutto scomparse, salvo quelle due o tre che resistono. Auguro a tutti i lettori italiani e stranieri una buona permanenza nei luoghi scelti per trascorrere il dovuto relax e divertimento... E al prossimo numero con un'altra ricca stagione di retro!

di **Daniele Brahimi**

GIUDIZIO FINALE

» Giocabilità 85%
Non difficile prendere la mano. Un po' più difficile vincere.

» Longevità 70%
Diverse opzioni renderanno il gioco a prova di polvere, poi il biliardo é il biliardo...

Scusate per il ritardo

Cari lettori, vogliamo innanzitutto ringraziarvi per la vostra pazienza e fedeltà nei confronti della nostra rivista.

Come molti di voi sanno, la creazione di contenuti di alta qualità richiede un impegno considerevole, e in questo periodo gli impegni della vita quotidiana hanno avuto un impatto significativo sulla nostra capacità di produrre dei contenuti. La gestione di impegni personali e professionali, insieme alle inaspettate sfide che si presentano ogni giorno, ha richiesto più tempo ed energia del previsto, portando inevitabilmente a un ritardo nella pubblicazione di questo nuovo numero di RetroMagazine World.

Nonostante le difficoltà, il nostro impegno per offrirvi articoli approfonditi e contenuti avvincenti rimane immutato. Abbiamo lavorato duramente per assicurarci che ogni pagina della rivista continui a riflettere la passione e la dedizione che ci contraddistinguono, e ci siamo presi questo tempo aggiuntivo per assicurarci che la qualità non ne risenta.

Allo stesso tempo voglio rassicurare tutti quelli che cominciavano a preoccuparsi del futuro di RetroMagazine World che la rivista e tutto quello che le ruota attorno sono vivi e vegeti. Vi assicuro che stiamo facendo tutto il possibile per mantenere fede al nostro impegno di continuare a offrirvi i contenuti che amate e che aspettate con entusiasmo.

Come di consueto, l'invito a partecipare alla vita della rivista con contenuti, domande, curiositá e quant'altro, é aperto a tutti. Vi aspettiamo!

Ancora una volta, vi ringraziamo per la vostra comprensione e supporto. Senza di voi, nulla di tutto ciò sarebbe possibile.
Siamo fiduciosi che troverete questo numero della rivista tanto interessante e stimolante quanto i precedenti, forse di piú, grazie al grande numero di recensioni di giochi.
Grazie per la vostra pazienza e il vostro continuo sostegno.

Francesco Fiorentini

Disclaimer

RetroMagazine World (fanzine aperiodica) è un progetto interamente no profit e fuori da qualsiasi circuito commerciale. Tutto il materiale contenuto è prodotto dai rispettivi autori e pubblicato grazie alla loro autorizzazione.

RetroMagazine World viene concessa al pubblico con licenza: Attribuzione - Non commerciale - Condividi allo stesso modo 4.0 Internazionale (CC BY-NC-SA 4.0 INT) https://creativecommons.org/licenses/by-nc-sa/4.0/deed.it

In pratica sei libero di: condividere, riprodurre, distribuire, comunicare o esporre in pubblico, rappresentare, eseguire e recitare questo materiale con qualsiasi mezzo e formato, modificare, rielaborare, trasformare il contenuto e basarti su di esso per altre opere, alle seguenti condizioni:

Attribuzione
Devi riconoscere una menzione di paternità adeguata, fornire un link alla licenza e indicare se sono state effettuate delle modifiche. Puoi farlo in qualsiasi maniera ragionevole possibile, ma non con modalità tali da suggerire che il licenziante avalli te o l'utilizzo del materiale da parte tua.

NonCommerciale
Non puoi utilizzare il materiale per scopi commerciali.

StessaLicenza
Se rielabori, trasformi il materiale o ti basi su di esso, devi distribuire i tuoi contributi con la stessa licenza del materiale originario.

Il licenziante non può revocare questi diritti fintanto che tu rispetti i termini della licenza.

Divieto di restrizioni aggiuntive
Non puoi applicare termini legali o misure tecnologiche che impongano ad altri soggetti dei vincoli giuridici su quanto la licenza consente loro di fare.

RetroMagazine World
Anno 8 - Numero 48 - LUGLIO 2024

Direttore Responsabile
Francesco Fiorentini
Vice Direttore
Marco Pistorio
Coordinatore Redazione/Editing
David La Monaca
Responsabili Area Web
Giorgio Balestrieri